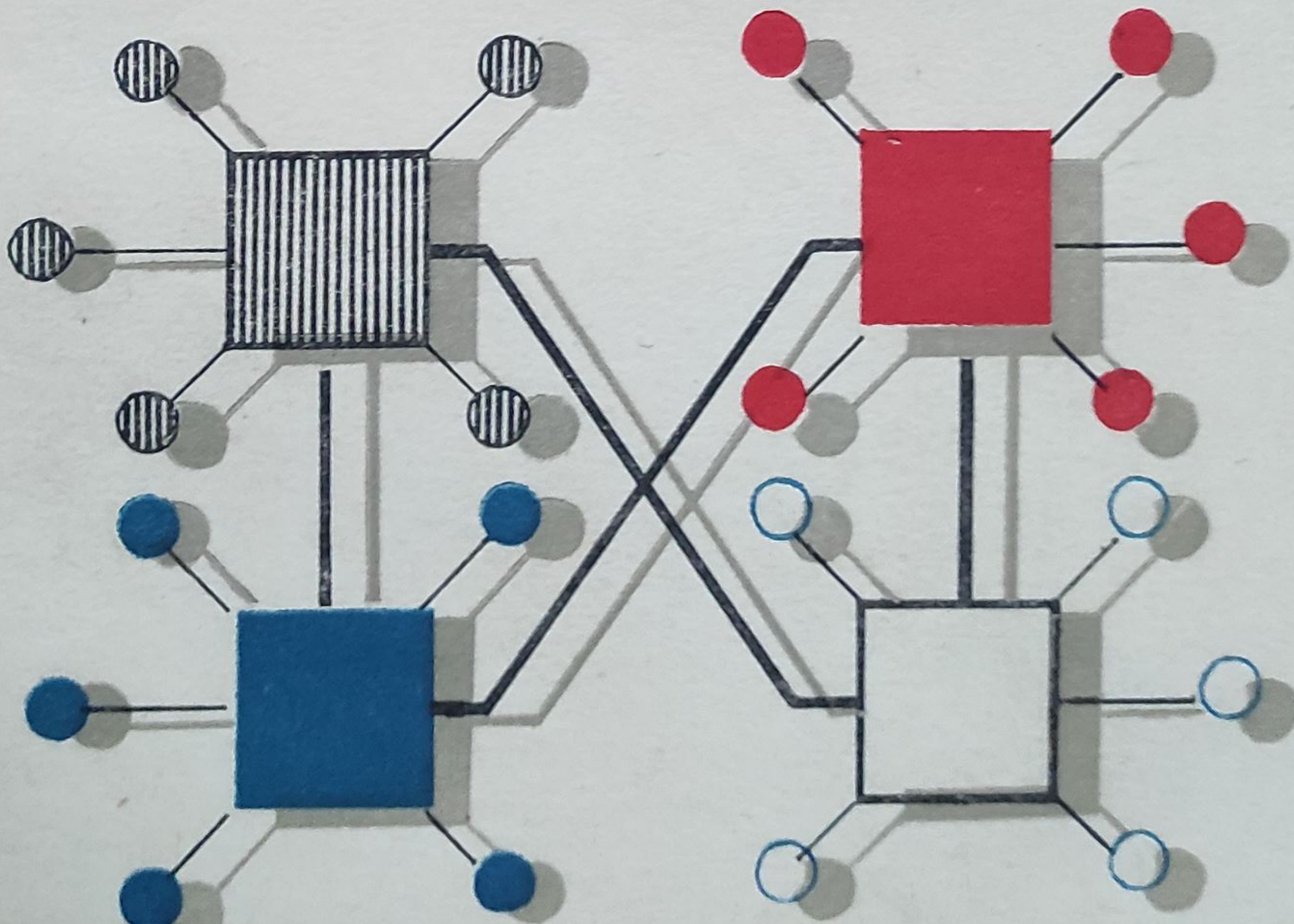
VASILE M. CĂTUNEANU • ANGELICA BACIVAROF

STRUCTURI ELECTRONICE DE ÎNALTĂ FIABILITATE

Toleranța la defectări



Prof. dr. ing. VASILE M. CĂTUNEANU Dr. ing. ANGELICA BACIVAROF

STRUCTURI ELECTRONICE DE ÎNALTĂ FIABILITATE

Toleranța la defectări

EDITURA MILITARĂ, BUCUREȘTI, 1989

PREFAȚA

Creșterea continuă a complexității sistemelor tehnice actuale și a importanței misiunilor pe care acestea trebuie să le îndeplinească impune implementarea unor metode eficiente de minimizare a riscului eventualelor defectări. În acest context, apariția unei lucrări de anvergură consacrate implementării toleranței la defectări — principala metodă ce poate fi utilizată pentru creșterea fiabilității sistemelor — este binevenită.

Importanța și actualitatea aspectelor abordate sînt evidențiate și de rezultatele unor studii de prognoză, care au stabilit că sistemele de mare răspundere funcțională ale anilor '90 vor avea inclusă toleranța la defectări drept una dintre caracteristicile lor principale.

Lucrarea de față reprezintă o contribuție de valoare a cercetării științifice românești la studiul aspectelor teoretice și practice ale realizării sistemelor de înaltă fiabilitate. Ea ocupă un loc privilegiat în literatura tehnică de specialitate românească și străină atît prin noutatea problematicii abordate, cît și prin echilibrul pe care reușește să-l păstreze între rigoarea teoretică și aplicabilitatea practică.

Un merit deosebit al volumului îl constituie abordarea complexă a modalităților de realizare a sistemelor de înaltă fiabilitate. Se remarcă analiza critică aprofundată a diferitelor metode de elaborare a sistemelor cu o anumită inteligență, aceea de a recunoaște și masca unele erori și de a-și realiza funcțiile de ieșire în mod corect. Se au în vedere nu numai aspectele tehnice ale metodelor de implementare a toleranței la defectări și influența acestora asupra performanțelor sistemelor, ci și implicațiile lor economice și sociale.

O atenție specială este acordată prezentării metodelor de realizare a toleranței la erori cu mijloace software, precum și a software-ului fiabil, cum sînt programarea N-versională sau programarea N-autotestabilă.

Un important merit al lucrării îl constituie dezvoltarea unor analize cantitative aprofundate referitoare la performanțele acestor sisteme, introducîndu-se în acest scop indicatori de fiabilitate specifici. De asemenea, atrag atenția studiile care privesc optimizarea diferitelor structuri tolerante la defectări.

Soluționarea eficientă a acestor analize este realizată prin utilizarea adecvată a unor metodologii bazate pe teoria probabilităților, algebra booleană, teoria grafelor, procesele markoviene etc.

Materialul expus în volum are la bază analiza critică a unei ample informații în domeniul abordat, în acest context contribuțiile originale ale autorilor — reliefate în mod gradat — avînd o pondere importantă.

Unitatea lucrării este asigurată de abordarea tematicii prin prisma parcurgerii etapelor realizării toleranței la defectări: identificarea automată a erorilor, mascarea lor, autoreconfigurarea sistemelor, realizarea unui software tolerant la erori și totodată evaluarea performanțelor sistemelor obținute. După opinia noastră este pentru prima oară cînd se abordează în mod aprofundat și unitar aspectele legate de corelația dintre fiabilitate și implementarea toleranței la defectări.

Lucrarea Structuri electronice de înaltă fiabilitate. Toleranța la defectări *constituie rodul unei îndelungate și rodnice activități didactice și științifice a autorilor, cunoscuți specialiști în domeniul fiabilității. Ea este caracterizată prin profunzimea studiilor, conduse pînă la nivelul finalizării practice; de aceea reușește să fie nu numai o carte de elevată ținută științifică, ci și o lucrare utilă celor interesați de soluții aplicative concrete.*

Volumul de față reprezintă un răspuns concret al cercetării științifice desfășurate în cadrul învățămîntului tehnic superior, la sarcinile puse în fața sa de către secretarul general al Partidului Comunist Român, tovarășul Nicolae Ceaușescu, în vederea realizării de sisteme cu performanțe tehnice și calitative superioare: „trebuie să împletim ferm cercetarea cu învățămîntul, cu producția, să legăm mai strîns întreaga cercetare de cerințele practice ale producției, de soluționarea problemelor care apar zi de zi în activitatea complexă pentru realizarea de produse mai bune, cu un nivel calitativ și tehnic tot mai ridicat"[1].

Competența autorilor și conținutul lucrării ne îndreptățesc convingerea că ea va deveni un instrument de lucru util specialiștilor din cercetare, proiectare și producție în vederea realizării unor sisteme cu risc minim al defectărilor.

Prof. dr.docent MIHAI DRĂGĂNESCU
Membru corespondent al Academiei
Republicii Socialiste România

---

[1] NICOLAE CEAUȘESCU, *Expunere la ședința comună a Plenarei Comitetului Central al Partidului Comunist Român, a organismelor democratice și organizațiilor de masă și obștești*, 28 noiembrie 1988, Editura Politică, București, 1988, p. 46.

Ridicarea continuă a nivelului tehnic și calitativ al produselor constituie pentru țara noastră o cerință fundamentală, pusă în fața întregii activități industriale. În acest sens, tovarășul Nicolae Ceaușescu, secretar general al Partidului Comunist Român, președintele Republicii Socialiste România, sublinia: „Trebuie să realizăm o asemenea dezvoltare a producției industriale încît nivelul ei tehnic și calitativ să fie competitiv și comparabil cu cele mai bune produse similare pe plan mondial“[1].

Direcția strategică vizînd asigurarea unei înalte fiabilități a produselor românești este vitală pentru sistemele de mare răspundere funcțională — sisteme de calcul, aeronautice, de energetică nucleară, militare, de dirijare a traficului — fiind deosebit de importantă pentru celelalte categorii de produse în vederea realizării unei competitivități sporite pe piața externă.

O direcție principală urmată pe plan mondial pentru asigurarea unei fiabilități ridicate în cazul sistemelor de mare răspundere funcțională — în special al celor informatice — constă în introducerea toleranței la defectări ca atribut arhitectural al acestora, prognozîndu-se, de altfel, că toleranța la defectări va constitui la începutul deceniului viitor una dintre caracteristicile principale ale acestei categorii de sisteme. Într-un asemenea context rezultă actualitatea problematicii toleranței la defectări, ea reprezentînd în același timp o importantă direcție de perspectivă pentru cercetare, proiectare și industrie.

Toleranța la defectări — atribut arhitectural al unui sistem, implementat la nivel hardware și/sau software — face posibilă îndeplinirea funcțiunilor acestuia chiar atunci cînd în sistem intervin una sau mai multe defectări.

Concept apărut cu peste două decenii în urmă, toleranța la defectări s-a dezvoltat ca domeniu de cercetare distinct după 1971, strîns legat de preocupările privind realizarea unor sisteme de calcul fiabile prin utilizarea unor structuri redondante.

Metodele toleranței la defectări nu vizează numai defectele fizice ale componentelor, ci și erorile de proiectare ale sistemului. Tehnicile de tolerare a erorilor, care pînă acum au avut un impact minor asupra sistemelor hardware,

---

[1] NICOLAE CEAUȘESCU, *Expunere la ședința comună a Plenarei Comitetului Central al Partidului Comunist Român, a organismelor democratice și organizațiilor de masă și obștești*, 28 noiembrie 1988, Editura Politică, București, 1988, p. 51.

sînt esențiale în primul rînd pentru sistemele informatice, la care software-ul este și va rămîne o parte critică, iar în al doilea rînd, pentru sistemele hardware proiectate și realizate în tehnologie VLSI.

Volumul este structurat în cinci capitole.

În primul capitol sînt prezentate problematica și conceptele de bază ale domeniului toleranței la defectări. Sînt evidențiate etapele unei strategii de implementare a toleranței la defectări, aprofundate în capitolele următoare în contextul cercetărilor pe plan mondial și proprii ale autorilor.

Capitolul 2 este destinat analizei structurilor redondante ce pot fi aplicate sistemelor hardware pentru protecția sistemului la apariția defectărilor. Sînt studiate în mod critic principalele scheme de implementare a redondanței, precum și problemele specifice legate de realizarea lor, cum ar fi acelea ale sincronizării elementelor replicate din structura sistemului. De asemenea, sînt evidențiate soluțiile optimale pentru implementarea diferitelor structuri redondante, propunîndu-se criterii de comparare a lor din punct de vedere calitativ; se menționează în special introducerea — în vederea evaluării globale a performanțelor unui anumit tip de redondanță — a indicelui de eficiență, care permite luarea în considerare simultană atît a efectelor utile ale aplicării redondanței prin creșterea fiabilității, cît și mărirea corespunzătoare a costului sistemului. Pe această bază sînt analizate în mod aprofundat structurile redondante de tip logică majoritară simplă, logică majoritară multiplă și de comutație, evidențiindu-se o serie de particularități ale implementării acestor structuri, cum ar fi — de exemplu — stabilirea nivelului optim de partiționare a sistemului pentru aplicarea redondanței.

Aceeași problemă a aplicării redondanței la nivel optimal este abordată și din alt punct de vedere: în ce condiții — pentru creșterea fiabilității unui sistem prin utilizarea aceleiași structuri redondante — trebuie să se procedeze la suplimentarea numărului de rezerve sau la modificarea gradului de partiționare a sistemului.

În capitolul 3 sînt studiate unele aspecte ce privesc tehnicile de detecție a defectărilor în sistemele tolerante la defectări atît în cazul funcționării off line *cît și în cel al funcționării* on line *a rezervelor. Astfel, în prima parte a capitolului se analizează principalele metode de generare a secvențelor de test, evidențiindu-se în mod critic particularitățile acestora; prin exemple și contraexemple se reliefează utilitatea sau unele utilizări necorespunzătoare ale acestor metode. În partea a doua se abordează într-o manieră unitară — pornindu-se de la un model general propus pentru aceste sisteme — o serie de structuri autotestabile. O atenție deosebită este acordată circuitelor de control care intră în componența ansamblelor de supraveghere a corectitudinii funcțiilor de ieșire ale sistemelor autotestabile. Se dezvoltă și se exemplifică metode destinate sintezei acestor circuite.*

Capitolul se încheie cu o trecere în revistă a principalelor metode de asigurare a unei testabilități ușoare pentru circuitele electronice corespunzătoare structurilor analizate.

Capitolul 4 cuprinde o analiză a unor tehnici sofisticate de realizare a toleranței la defectări, cum ar fi cele bazate pe autoreconfigurare și cele implementate cu mijloace software. Se menționează în mod special modelul reconfigurării optimale în sistemele multiprocesor, precum și studiile privind realizarea software-ului tolerant la erori, utilizîndu-se metodele blocurilor de resta-

bilire, programarea N-versională și programarea N-autotestabilă. În încheiere sînt analizate arhitectura și caracteristicile a șase sisteme tolerante la defectări, evidențiindu-se pentru fiecare tehnicile abordate.

Ultimul capitol este consacrat reliefării performanțelor de fiabilitate ale unor structuri tolerante la defectări, luîndu-se în considerare defectările posibile. În prima parte sînt dezvoltate în principal modele pentru analiza fiabilistică a structurilor redondante utilizate cu precădere în sistemele digitale în prezența unor defectări de tip permanent, fără a lua în considerare posibilitățile de reparare. Apoi, cu aceeași ultimă ipoteză, se elaborează o metodică de analiză a probabilității bunei funcționări în prezența defectărilor cvasitemporare. Ultima parte a analizei se referă la sistemele reparabile. Folosindu-se modelele markoviene se studiază — pe baza unor studii de caz — performanțele de fiabilitate ale acestor sisteme.

Volumul este bazat în principal pe cercetările autorilor în acest domeniu, dar și pe analiza unei largi bibliografii, îndeosebi articole publicate în principalele reviste de specialitate — IEEE Transactions on Computers, IEEE Transactions on Reliability etc. —, precum și în volumele unor congrese internaționale de specialitate — în principal International Symposium on Fault-Tolerant Computers ș.a.

Lucrarea se adresează specialiștilor din cercetare, proiectare, producție din domeniile tehnicii de calcul, nuclear, aeronautic, comunicații, militar tuturor celor care se ocupă de sisteme complexe de mare răspundere funcțională, precum și studenților din învățămîntul tehnic superior și doctoranzilor în specialitățile amintite.

AUTORII

CAPITOLUL 1

PRINCIPII GENERALE PRIVIND SISTEMELE TOLERANTE LA DEFECTĂRI

1.1. INTRODUCERE

Aspectele referitoare la elaborarea unui sistem sigur în funcționare pot fi abordate în două moduri de lucru complementare: *evitarea defectării sistemului* și, respectiv, *toleranța la defectări a sistemului*.

Evitarea defectării sistemului constă în încercarea — realizată încă din faza de proiectare a acestuia — de eliminare sau limitare a funcționării anormale prin utilizarea de componente fiabile, regimuri de lucru facile pentru componentele sistemului, eliminarea erorilor de concepție cu diferite tehnici de verificare a proiectării logice sau programe de verificare a funcționării. Dar, cu toată adoptarea tehnicilor de evitare a defectărilor, acestea sînt în mod uzual prezente în sistemul construit; de exemplu, din cauza complexității pot apărea defectele de proiectare, iar îmbătrînirea (uzura) componentelor hardware ale sistemului poate conduce la defectarea acestuia.

Toleranța la defectări a unui sistem este un atribut arhitectural al acestuia, care face posibilă operarea sistemului chiar atunci cînd în structura sa intervin unul sau mai multe defecte. În lipsa acestei calități, sistemul ar avea o funcționare anormală. Toleranța la defectări se realizează cu o suplimentare a părții hardware și/sau software din structura sistemului, care va acționa atunci cînd are loc un defect în sensul păstrării funcțiilor sistemului, fie prin mascarea defectărilor care apar, fie prin detecția defectărilor și reconfigurarea corespunzătoare a sistemului.

Tehnicile de tolerare a defectărilor au fost inițial dezvoltate în legătură cu protejarea sistemului construit cu componente hardware nefiabile. O dată cu utilizarea componentelor semiconductoare și mai ales a tehnologiilor integrate, tehnicile de tolerare a defectărilor au fost dezvoltate pentru aplicații speciale ale sistemelor care nu admiteau întreruperi în îndeplinirea misiunii lor.

Există cîteva argumente care pledează pentru implementarea toleranței la defectări mai pregnant în domeniul sistemelor informatice (sisteme de comunicație, procesoare, calculatoare etc.), care au devenit elemente de analiză și decizie în cele mai diverse domenii. Preluarea de către sistemul

informatic a funcțiilor complexe, de la rezolvarea programelor uzuale ale unor activități de rutină pînă la problemele de esență în conducerea și controlul proceselor industriale, impune un nivel de fiabilitate ridicat.

Analiza aprofundată a rezultatelor obținute privind fiabilitatea și mentenabilitatea unor sisteme de calcul [5], [27], [36], [22] relicfează că timpul necesar detecției defectelor și repunerii sistemului în funcțiune este în unele cazuri nepermis de mare, conducînd la scăderea disponibilității sistemului, ceea ce sugerează că implementarea toleranței la defectări — mai ales în cazul sistemelor mari — poate conduce la rezultate mai bune de disponibilitate, securitate, inclusiv din punctul de vedere al costului, comparativ cu realizarea sistemelor de înaltă fiabilitate, dar intolerante la defectări. Astfel, apare contradictoriu faptul că un sistem de calcul, care conține sisteme de operare sofisticate pentru rezolvarea automată a unor serii mari de probleme complicate, să fie indisponibil un timp în care ar putea prelucra milioane de operații, din cauza unei componente defecte, unei conexiuni desprinse sau unei celule mai lente decît celelalte zeci de mii de componente ale memoriei principale, de exemplu. Soluționarea acestei clase de probleme de „supraviețuire" a sistemului cu ajutorul uneia dintre tehnicile toleranței la defectări nu este mai dificilă decît realizarea componentelor și sistemelor cu o rată a defectărilor foarte scăzută.

Desigur, argumentul de bază în favoarea utilizării toleranței la defectări ar putea fi *reducerea costului* necesar asigurării — pentru sistemul tolerant la defectări — unui timp de bună funcționare echivalent cu cel al sistemului foarte fiabil, dar netolerant la defectări. Evident, introducerea toleranței la defectări ca o caracteristică a sistemului va însemna o investiție inițială — uneori substanțială — necesară proiectării sistemului și cantității suplimentare de hardware sau/și software. Totuși, o analiză globală comparativă evidențiază pentru sistemele cu structură tolerantă la defectări — o reducere a costului necesar obținerii unui anume timp de viață al sistemului; aceasta atît datorită costului mai scăzut al componentelor sistemului, care nu trebuie să fie la fel de fiabile ca acelea utilizate în sistemele de mare siguranță în funcționare, dar intolerante la defectări, cît și — în special — datorită reducerii sau chiar eliminării timpului necesar mentenanței corective.

Mai mult, toleranța la defectări a sistemelor este o necesitate în cîteva domenii de aplicații, cum ar fi:

- domenii în care o funcționare anormală a sistemului *poate pune în pericol viața omului* — controlul și desfășurarea traficului aerian, feroviar sau alte forme ale transportului cu vehicule, controlul proceselor din centralele nucleare, monitorizării în spitale etc.;
- domenii în care o întrerupere neprevăzută a sistemului cauzează *pierderi importante financiare*, ca de exemplu controlul proceselor din industria chimică, metalurgică, controlul sistemelor de comutație telefonică, sau alte tipuri de comunicații, controlul distribuției energiei electrice în sistemul energetic național etc.;
- domenii în care nu este posibil accesul în vederea mentenanței manuale a sistemului, cum ar fi echipamente ale sateliților lansați pe orbite, sau ale stațiilor subacvatice etc.

În plus, la aceste argumente în favoarea utilizării sistemelor tolerante la defectări se mai adaugă suportul psihologic adus utilizatorului uman care

depinde sau interacționează cu astfel de sisteme. Atunci cînd defectarea unui sistem poate afecta securitatea unor persoane sau aduce pagube materiale importante, utilizarea toleranței la defectări reduce grija avută pentru apariția primului defect într-un element al sistemului — defect despre care se cunoaște că ar avea consecințe catastrofale — chiar dacă probabilitatea apariției sale este de ordinul 10^{-6} pe zi sau chiar mai mică. Aceasta face ca sistemele tolerante la defectări să fie preferabile — din punctul de vedere al utilizatorilor — sistemelor foarte fiabile, dar netolerante la defectări.

Deși au existat unele preocupări în vederea construirii sistemelor digitale tolerante la defectări încă din perioada anilor '50 [35], problema a început să fie pusă sistematic de abia după 1960, strîns legată de programele zborurilor interplanetare și de aplicațiile calculatoarelor în timp real, cînd intră în discuție în primul rînd securitatea vieților umane. Cercetări în acest sens au fost desfășurate în S.U.A. [3], [1], [8], [36], [34], Franța [22], [23], [14], U.R.S.S. [26], R. F. Germania, Anglia [2], Japonia [29], [32] ș.a.

Cercetările de implementare a toleranței au fost intensificate după 1971 îndeosebi în domeniul calculatoarelor, semnificative în acest sens fiind înființarea Comitetului tehnic pentru calculatoare tolerante la defectări din cadrul Societății pentru calculatoare a I.E.E.E., ca și lucrările simpozionului internațional „Fault Tolerant Computing" [36]. Acesta din urmă a devenit principalul forum internațional unde au fost prezentate cele mai noi idei în proiectarea sistemelor tolerante la defectări, în modelarea și analiza sistemelor, în testarea și diagnoza defectărilor sistemelor ca și în alte domenii conexe.

1.2. STRUCTURA SISTEMULUI ȘI IMPLEMENTAREA TOLERANȚEI LA DEFECTĂRI

Noțiunea de sistem este foarte generală, ea fiind utilizată în cele mai diverse domenii. Contextul lucrării de față se referă la sistemele tehnice, în mod particular la sistemele de calcul.

Conform teoriei generale a sistemelor, un sistem fizic real este descris printr-un ansamblu de atribute măsurabile, numite variabile accesibile, care reprezintă singurele sale legături cu exteriorul. Cu ajutorul acestor variabile se definește [33] noțiunea de *obiect abstract* ca o mulțime de perechi ordonate de funcții de timp:

$$\mathcal{A} = \{(\mathbf{X}(t_0, t_1),\ \mathbf{Y}(t_0, t_1))\};\ t_0, t_1 \in (o, \infty) \tag{1.1}$$

care satisfac condiția de concatenaritate.

Definiția dată obiectului abstract este o exprimare formală a faptului că orice interacțiune cu un obiect fizic implică varierea unora dintre atributele acestui obiect și observarea variațiilor care rezultă la celelalte atribute. Atributele care sînt variate joacă rolul de intrări (cauze), iar variațiile rezultate sînt ieșirile (efecte). Numind pe $\mathbf{X} = (x_1, x_2 \ldots, x_n)$ vector de intrare și pe $\mathbf{Y} = (y_1, y_2, \ldots, y_m)$ vector de ieșire, rezultă că obiectul abstract este în esență o mulțime de perechi intrare-ieșire.

Un sistem real poate fi modelat printr-un singur obiect abstract dacă analiza acestuia este situată la nivel global, atunci cînd se consideră numai

variabilele de intrare și ieșire, dar poate fi modelat printr-o interconexiune de obiecte abstracte, dacă este analizat la nivel structural, considerîndu-se elementele sale componente.

Astfel, se consideră o mulțime de obiecte abstracte $\mathcal{A}_1, \mathcal{A}_2, \ldots \mathcal{A}_N$, în care o parte dintre intrările sau ieșirile lui $\mathcal{A}_i$ pot fi constrînse să fie egale — pentru toate valorile lui i — cu unele intrări sau ieșiri ale altor obiecte abstracte din mulțime. O astfel de combinație de obiecte abstracte formează un sistem privit structural.

În figura 1.1 se dă un exemplu de combinație a două obiecte abstracte. Variabilele de ieșire Y_1 ale lui $\mathcal{A}_1$ sînt variabile de intrare X_2 pentru $\mathcal{A}_2$ și variabile de stare pentru $\mathcal{A}$. Rezultă astfel structura sistemului $\mathcal{A}$. La rîndul lor, $\mathcal{A}_1$ sau $\mathcal{A}_2$ pot fi prezentate ca o combinație de obiecte abstracte componente, avînd astfel exprimată structura sistemului la un alt nivel.

Acest model general, furnizat de teoria sistemelor, poate fi utilizat în vederea descrierii unui sistem real sub diferite aspecte, depinzînd de interpretarea fizică dată variabilelor de intrare, ieșire și stare. În particular, este necesar să se cunoască structura sistemului atunci cînd se urmărește realizarea unor sisteme fiabile, deoarece structura unui sistem are un impact major asupra fiabilității sale; analiza și optimizarea corelației structură-fiabilitate este de o importanță deosebită în cazul proiectării sistemelor tolerante la defectări.

Analiza performanțelor de fiabilitate ale unui sistem poate fi abordată global sau la nivelul structurii interne a sistemului.

Dacă sistemul este privit global — în absența informațiilor legate de structura sa internă — pot fi elaborate modele de fiabilitate care permit, de exemplu, determinarea relațiilor funcționale între performanțele sistemului — variabilele de ieșire — și solicitări — variabilele de intrare. Atunci cînd poate fi evidențiată structura sistemului, este posibilă elaborarea unor modele de fiabilitate care leagă performanțele sistemului de solicitări prin intermediul parametrilor elementelor componente ale sistemului; deși mai laborioase, acestea prezintă avantajul că permit o modelare fiabilistică mai aprofundată, devenind posibilă pe această bază și optimizarea structurii sistemului din punctul de vedere al fiabilității.

Implementarea oricărei tehnici de toleranță la defectări implică o intervenție în structura sistemului — care trebuie să fie bine controlată —

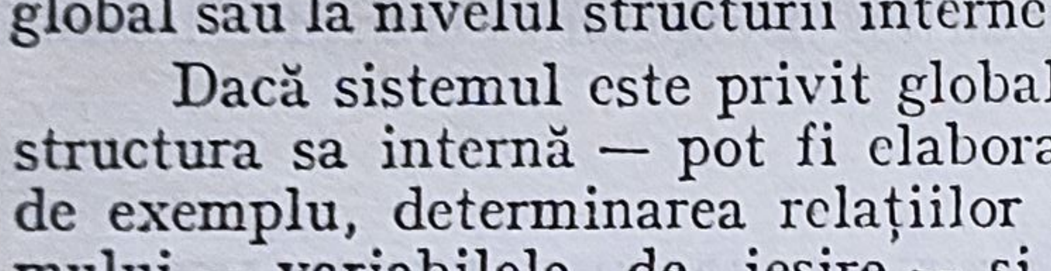


Fig. 1.1. Reprezentarea abstractă a unui sistem.

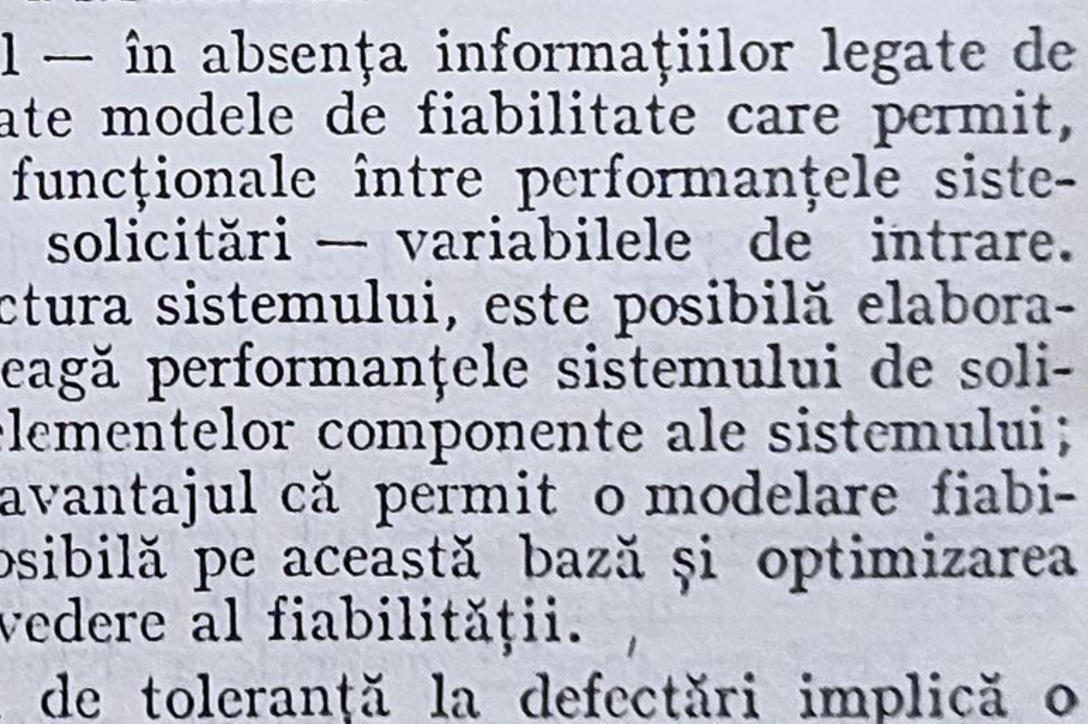


Fig. 1.2. Reprezentarea abstractă a unui sistem de calcul.

la un anumit nivel. Așadar, structura sistemului va sta la baza metodologiei de proiectare și realizare a toleranței la defectări.

Definițiile date mai sus sistemului și structurii sale sînt suficient de generale, fiind aplicabile oricărui sistem real; în particular, ele sînt utilizabile și în cazul sistemelor *software*.

Activitatea unui sistem software nu este generată de componente fizice, ci de componente abstracte, numite procese. Activitatea unui proces este gîndită ca secvența acțiunilor generate de un program sau un set de programe rulate. Un sistem software poate fi considerat [2] ca un singur proces compresiv, care — în general — poate fi structurat ca un set de procese interactive, analog componentelor sistemului.

În practică însă, software-ul nu poate fi considerat izolat de hardware. Interacțiunea între cele două componente, software și hardware, într-un sistem poate fi modelată în diferite moduri. În continuare, se va prezenta un astfel de model evidențiindu-se ponderea componentei software într-un sistem de calcul.

Se consideră un sistem de calcul, care privit ca un sistem hardware, are patru componente: un cititor de cartele, o imprimantă, o unitate de memorie și un procesor. În memorie este stocat un singur program și datele sale, acesta fiind executat de către procesor.

Privit ca un sistem software, acest sistem de calcul poate fi modelat ca un proces secvențial, cuprinzînd activitatea generată de execuția programului. Pentru a analiza structura internă a procesului, este necesar să se considere relația de interacțiune hardware-software.

Din punctul de vedere al hardware-ului, software-ul există numai ca o parte a stării unității de memorie și rămîne pasiv, iar programele diferă numai datorită valorilor datelor. O examinare a componentei hardware care înmagazinează componenta software evidențiază că software-ul este reprezentat ca o secvență adresabilă de cuvinte de memorie, o structură validă, dar puțin utilă în practică. Un studiu mai detaliat al componentei hardware ar conduce pînă la organizarea memorării la nivel de bit. Un asemenea mod de a examina structura software-ului nu este relevantă într-o analiză de fiabilitate a sistemului de calcul sau pentru implementarea toleranței la defectări.

Pornind de la definiția dată pentru sistem — ca un obiect abstract — se propune următoarea abordare în cazul sistemului analizat: programul stocat în memorie este privit ca un obiect abstract, fiind o componentă a sistemului; celelalte componente ale sistemului sînt obiectele abstracte manipulate de program, atunci cînd acesta este rulat. Cu punctul de vedere menționat, sistemul considerat are structura indicată în figura 1.2. Se observă că programul nu mai apare inclus în memoria procesorului,ci devine o componentă a structurii sistemului.

În această reprezentare abstractă a sistemului de calcul, componentele sale corespund componentelor hardware ale sistemului fizic: cititorul de cartele și imprimanta sînt identice cu componentele fizice cititor cartele și imprimantă ale sistemului; datele reprezintă programul stocat în memorie; procesorul corespunde funcțiilor aritmetice și logice executate de procesorul material al sistemului de calcul. Se remarcă totuși că procesorul abstract

prezentat în acest model nu mai execută programul: programul este privit ca obiectul abstract care controlează interacțiunile dintre componentele abstracte ce intră în structura sistemului.

Structura internă a componentelor abstracte poate fi modelată în continuare examinînd, în același mod, structura internă a componentelor hardware.

Pentru partea de software se acceptă o divizare constînd din componentele instrucțiuni de bază. În acest sens, programul poate fi reprezentat ca un arbore avînd originea (rădăcina) programul propriu-zis, ramurile corespunzînd instrucțiunilor individuale conținute în program. O astfel de structură arborescentă are avantajul că evidențiază modul de descompunere ierarhică a unui program. Totuși, un astfel de model este util numai în măsura în care clarifică aspectele organizării unui program, precum și în implementarea tehnicilor de proiectare pentru sistemele software cum sînt tehnica descendentă „top-down", sau tehnica ascendentă „bottom-up". Această structură este — se pare — mai utilă pentru programele scrise în limbaje de nivel înalt, care favorizează modularitatea.

Modelul abstract — reprezentat grafic în figura 1.2 — propus pentru sistemul analizat permite construirea unui model de fiabilitate al sistemului, evidențiind în mod realist ponderea componentei software în evoluția sistemului.

1.3. CONCEPTE DE BAZĂ

În cele ce urmează sînt introduse cîteva concepte de bază utilizabile în domeniul sistemelor tolerante la defectări.

Definiția cea mai acceptată dată *sistemelor tolerante la defectări* este aceea potrivit căreia sistemul își poate continua execuția corectă a funcțiilor sale de intrare/ieșire, fără o intervenție din exterior, în prezența unei anumite mulțimi de defectări ce apar în timpul funcționării sale [5], [18], [9]. Definiția este aplicabilă la orice nivel al unui sistem, indiferent dacă această caracteristică este implementată hardware sau software.

Prin *execuție corectă* se va înțelege că rezultatele operării sistemelor nu conțin erori, iar timpul de execuție al unei operații nu depășește o limită specificată.

O defectare ce apare în timpul funcționării *operaționale* a unui sistem este o schimbare în valoarea uneia sau a mai multor variabile de ieșire sau de stare ale sistemului; defectarea constituie consecința imediată a evenimentului defect.

Evenimentul fizic *defect* este o imperfecțiune fizică a unui element al sistemului, care antrenează o *funcționare permanent, temporar sau intermitent eronată* sau poate fi *un factor extern* care conduce la nefuncționarea sistemului.

Defectările operaționale ale unui sistem se clasifică după *durată* în permanente și temporare, iar după *extindere* în singulare și multiple.

Defectările permanente sînt cauzate de defectele permanente ale componentelor sistemului. Protecția împotriva acestora pentru realizarea execuției corecte a funcțiilor sistemului impune fie prezența componentelor de rezervă,

care vor înlocui componentele defecte, fie măsuri de reconfigurare pentru continuarea operațiilor, fără a mai interveni componenta defectă în funcționarea sistemului.

Defectele *temporare* sînt de durată limitată, fiind cauzate de nefuncționările temporare ale componentelor sau de interferențe externe. Aceste defectări pot fi caracterizate prin parametrul specific *durată maximă*; defectările care au acest parametru de valoare mare vor fi interpretate de algoritmii de protecție ai sistemului în vederea realizării toleranței la defectări ca fiind permanente [21], utilizîndu-se metodele de protecție adecvate pentru fiecare caz în parte.

Defectările *pseudotemporare* sînt defectările cauzate de defectele permanente ale componentelor sistemului, pentru a căror manifestare este necesară o anume combinație a valorilor semnalelor de intrare sau variabilelor logice ce caracterizează stările sistemului. Sînt specifice în general circuitelor logice.

Clasificarea corespunzătoare extinderii influenței unui defect este aplicabilă atît defectărilor permanente, cît și defectărilor temporare. Defectările *locale* (*singulare*) sînt acele defectări care afectează numai o singură variabilă corespunzătoare unui semnal de ieșire sau unei stări interne, în timp ce defectările *distribuite* (*multiple*) sînt acele defectări care afectează mai multe variabile de ieșire sau de stare ale sistemului. Prezența în structura sistemelor a circuitelor integrate de tip LSI sau VLSI face ca defectările multiple să fie mai întîlnite decît defectările singulare. Defectările *multiple* pot fi cauzate de defectele singulare ale cîtorva elemente critice ale sistemului, ca de exemplu — în cazul sistemelor de calcul — sursele de putere, generatoarele de tact sau magistralele de date.

Eroarea este un simptom al unui defect, fiind cauzată de prezența unei defectări la locul de acțiune a defectului.

Definiția dată mai sus pentru sistemele tolerante la defectări se bazează pe ipoteza că *defectările de proiectare* ale sistemului au fost eliminate înainte de a începe utilizarea acestuia. *O interpretare* mai *generală* dată toleranței la defectări presupune ca aceasta să includă și abilitatea tolerării defectărilor de proiectare, nedetectate înainte de utilizarea sistemului. Tolerarea defectelor este privită nu în sensul amînării defectării sistemului, ci a luării măsurilor de precauție în vederea localizării automate a elementelor defecte și dezactivării hardware-ului sau software-ului afectat de erori, sistemul continuîndu-și funcționarea pe baza elementelor redondante. Pentru ca sistemul să fie inclus în sfera sistemelor tolerante la defectări este necesar ca această dezactivare să fie realizată *automat* de către sistemul în cauză. Într-un sens strict, un sistem este tolerant la defectări numai dacă nu este necesară o intervenție externă care să-i implementeze această caracteristică.

Defectările de *proiectare* sînt consecința unor specificații de proiectare incomplete sau incorecte; pot avea loc atît în partea de hardware a sistemului, cît și în partea de software a acestuia.

Un sistem cu un grad de tolerare a defectărilor care nu îndeplinește toate condițiile definiției sistemelor tolerante la defectări poate fi considerat ca fiind *parțial tolerant la defectări* [5]. Astfel, un sistem de calcul parțial tolerant la defectări „fail-softly", este capabil de a-și reduce capaci-

tatea de calcul specifică la apariția unei defectări, devenind prin reconfigurare un „sistem mai mic", sau de a-și încetini viteza specifică de execuție etc.

De menționat că, cele mai multe sisteme actuale, fără a fi tolerante la defectări folosesc cîteva tehnici de tolerare a defectărilor, cum ar fi în cazul sistemelor de calcul: microdiagnozele [9], [18], testarea la paritate [17], dispozitivele de supraveghere interne etc.

Rezultă deci că tolerarea la defectări este un aspect esențial al realizării sistemelor fiabile și în sensul cel mai general înseamnă *execuție corectă a unui set de operații specificate în prezența unor defecte*. Fiabilitatea sistemului* este astfel asigurată, în general, prin utilizarea redondanței structurale, care asigură funcționarea neîntreruptă a sistemului, iar disponibilitatea** și prin testarea sistemului în timpul funcționării normale, fără a mai exista timpi de întrerupere pentru mentenanță.

Atunci cînd, pentru a se implementa toleranța la defectări unui sistem, se utilizează forme ale redondanței de tip static, se spune că sistemul prezintă o *toleranță statică la defectări*, iar cînd se utilizează diferite procedee de detecție a defectărilor sistemului, urmate de reconfigurări automate ale acestuia, sistemul va fi cu *toleranță dinamică la defectări*.

1.4. MODELAREA CONCEPTULUI DE TOLERANȚĂ LA DEFECTĂRI

În cele ce urmează se va dezvolta un model pentru analiza conceptului de toleranță la defectări a sistemelor, evidențiindu-se în aceeași manieră și conceptul de diagnozabilitate. Modelul are un înalt grad de generalitate, ceea ce face ca el să fie aplicabil atît sistemelor hardware cît și sistemelor software. Modelul permite reprezentarea efectelor erorilor de proiectare sau defectelor fizice ale componentelor sistemului.

1.4.1. SISTEM CU DEFECTE

Se definește un *sistem cu defecte* ansamblul format din sistemul respectiv, mulțimea de defecte potențiale ale sistemului și descrierea a ceea ce se întîmplă în sistemul originar ca rezultat al acțiunii fiecărui defect.

În continuare, cu tripletul $(\mathcal{S}_p, \mathcal{R}, \rho)$ se definește formal o schemă de reprezentare a unui sistem, unde: $\mathcal{S}_p$ este clasa specificațiilor sistemelor; $\mathcal{R}$ — clasa realizărilor sistemelor, iar $\rho : \mathcal{R} \to S_p$ — funcția de realizare a specificațiilor; dacă $R \in \mathcal{R}$ atunci $\rho(R) \in \mathcal{S}_p$.

---

* Fiabilitatea reprezintă capacitatea unui sistem de a-și îndeplini funcția pentru care a fost proiectat și realizat. Se măsoară prin funcția de fiabilitate, $R(t)$, definită ca fiind probabilitatea ca sistemul să funcționeze fără defectări în intervalul de timp $(0, t)$ în condiții determinate.

** Disponibilitatea reprezintă capacitatea unui sistem de a-și îndeplini funcția pentru care a fost proiectat și realizat în condiții de utilizare precizate la un moment de timp dat. Se măsoară prin funcția de disponibilitate, $A(t)$, definită ca probabilitatea ca sistemul să fie în stare de funcționare în momentul de timp t.

Prin clasă de sisteme se înțelege, în acest context, o clasă de sisteme formale, adică o mulțime de structuri de același tip specificate în mod formal, fiecare avînd o comportare asociată determinată de tipul structurii. Exemple în acest sens sînt mulțimea programelor formale relative la o metodă de programare, sau toate sistemele automate finite, toate rețelele de comutație etc.

Într-o reprezentare $(\mathcal{S}_p, \mathcal{R}, \rho)$, fiecărei realizări individuale $R \in \mathcal{R}$ îi este repartizată o specificație $\rho(R) \in \mathcal{S}_p$, cu interpretarea că $\rho(R)$ este o reprezentare structurală a unui sistem definit de R. De exemplu, dacă $\mathcal{S}_p$ este mulțimea tuturor mașinilor secvențiale și $\mathcal{R}$ este mulțimea tuturor rețelelor de comutație secvențiale, atunci $\rho(R)$ este o mașină secvențială definită de rețeaua de comutație R.

Dacă $\mathcal{S}_p = \mathcal{R}$, ρ este funcția identitate pe $\mathcal{R}$, iar schema $(\mathcal{R}, \mathcal{R}, \rho)$ va fi notată mai simplu $\mathcal{R}$.

Pe baza acestei definiții formale, se va defini în continuare conceptul de sistem cu defect.

Unui sistem cu defecte, într-o reprezentare $(\mathcal{S}_p, \mathcal{R}, \rho)$, îi corespunde o structură de tipul (S, F, Φ), unde $S \in \mathcal{S}_p$, F este o mulțime de defecte ale lui S, iar $\Phi : F \to \mathcal{R}$, astfel că pentru oricare $f \in F$, $\Phi(f) = S^f$, în care S^f reprezintă sistemul cu defectul f. În acest context, se poate scrie că atunci cînd $f \in F$, sistemul S^f, $S^f = \Phi(f)$, este rezultatul existenței lui f.

Deci, un sistem cu defecte este o reprezentare formală a unui sistem potențial nefiabil: S reprezintă sistemul fără defecte, F — o mulțime de defecte potențiale care se pot produce în procesul realizării lui S, iar pentru un defect f, $f \in F$, $S^f = \Phi(f)$ este realizarea sistemului în prezența lui f.

Defectul f poate fi un *defect impropriu* dacă procesul de realizare a lui S în prezența lui f conduce la sistemul specificat S, adică $\rho(S^f) = S$.

Un defect f este *propriu*, dacă procesul de realizare a lui S în prezența defectului f nu mai conduce la realizarea lui S, adică $\rho(S^f) \neq S$.

Un aspect important al formalismului dezvoltat mai sus, este acela că acest formalism permite reprezentarea oricărui tip de defect al sistemului: defect de proiectare sau defect al unei componente fizice a sistemului, care apare în timpul vieții acestuia datorită unor cauze aleatoare sau uzurilor. Mai mult, formalismul evidențiază distincțiile între cele două tipuri de defecte.

În formalismul $(\mathcal{S}_p, \mathcal{R}, \rho)$ unde $\mathcal{S}_p$ este clasa specificațiilor, iar $\mathcal{R}$ — clasa realizărilor sistemelor ca rezultat al procesului de proiectare, prezența defectelor de proiectare este indicată de relația $\mathcal{S}_p \neq \mathcal{R}$, adică specificațiile diferă de realizări. Prin urmare, dacă (S, F, Φ) este un sistem cu defecte, F reprezintă o mulțime a defectelor posibile de proiectare atunci cînd $S_p \neq \mathcal{R}$.

Pe de altă parte, în reprezentarea defectelor componentelor fizice care apar în timpul vieții utile ale unui sistem, tripletul $(\mathcal{S}_p, \mathcal{R}, \rho)$ este de tipul $(\mathcal{R}, \mathcal{R}, \rho) = \mathcal{R}$, deoarece se presupune absența erorilor de proiectare, iar dacă (S, F, Φ) este un sistem cu defecte, F este mulțimea defectelor componentelor fizice din timpul vieții utile a sistemului.

1.4.2. TOLERANȚĂ LA DEFECTĂRI ȘI DIAGNOZABILITATE

Se consideră un sistem cu defecte (S, F, Φ) într-o schemă de specificare $(\mathcal{S}_p, \mathcal{R}, \rho)$ și un defect impropriu, f, $f \in F$. Se pune problema dacă sistemul $S^f = \Phi(f)$ este utilizabil, în sensul că funcționarea lui se aseamănă — în

limite acceptabile — cu funcționarea unui sistem fără defecte. În caz afirmativ, înseamnă că există relația de toleranță a defectului f, acceptată de sistemul S.

Se definește în acest fel o *relație de toleranță la defectări* pentru o schemă de reprezentare de tipul $(\mathcal{S}_p, \mathcal{R}, \rho)$ ca fiind o relație structurală τ între $\mathcal{R}$ și $\mathcal{S}_p$, $\tau \subset \mathcal{R} \times \mathcal{S}_p$, astfel că pentru oricare $R \in \mathcal{R}$, mulțimea $(R, \rho(R)) \in \tau$, deoarece $\rho \subset \tau$.

Condiția $(R, \rho(R)) \in \tau$, pentru oricare $R \in \mathcal{R}$, reflectă cerința că funcționarea unei realizări a sistemului prezintă întotdeauna toleranță la defectări.

Cu acest formalism se poate da în continuare următoarea definiție:

Dacă (S, F, Φ) este un sistem cu defecte într-o reprezentare $(\mathcal{S}_p, \mathcal{R}, \rho)$, τ este o relație de toleranță pentru defectul f, $f \in F$, dacă $(\Phi(F), S) \in \tau$, iar atunci cînd mulțimea $(\Phi f), S) \neq \tau$, defectul este diagnosticat cu relația structurală τ.

Astfel, se poate nota că un defect impropriu este întotdeauna acceptat de sistem deoarece $\rho(\Phi)(f)) = S$, de unde rezultă $(\Phi(f), S) \in \tau$.

Pe de altă parte, dacă un defect f este diagnozabil cu relația τ, înseamnă că va exista o secvență de semnale de intrare care va produce o ieșire eronată, prin aceasta rezultînd un test de diagnosticare a defectului f.

Cele două concepte — toleranță la defectări și diagnosticare a defectelor — sînt complementare; dar, este posibil ca în cazul unor sisteme de tipul (S, F, Φ) să existe o relație τ de toleranță a defectărilor și o relație τ' de diagnosticare a defectelor, astfel încît $\tau \neq \tau'$.

1.5. STRATEGII DE IMPLEMENTARE A TOLERANȚEI LA DEFECTĂRI

Relația structurală τ — din modelul anterior — care asigură sistemului toleranța la defectări poate fi o relație de mascare a defectărilor sau de corecție și reconfigurare adecvată a sistemului. În general se disting trei tipuri de strategii ce pot fi urmate pentru implementarea toleranței la defectări:

— strategii bazate pe diagnosticarea defectărilor și înlocuirea elementelor defecte;

— strategii bazate pe mascarea defectărilor;

— strategii hibride bazate pe mascarea defectărilor, diagnosticarea și înlocuirea elementelor defecte. În figura 1.3. se indică sintetic aceste strategii, reliefîndu-se mecanismul tolerării defectărilor. În toate cazurile intervine *redondanța*: fie o redondanță la nivel hardware sau software, fie o redondanță a funcțiilor sistemului.

O metodologie de implementare a toleranței la defectări aplicată unui sistem trebuie să înceapă cu specificarea cerințelor de fiabilitate pentru acel sistem, să continue cu selecția metodelor de diagnosticare a defectărilor și a algoritmilor de mascare a defectărilor sau reconfigurare a sistemului și se termine cu evaluarea toleranței la defectări — a relației structurale τ — și a performanțelor de fiabilitate și economice aduse sistemului. În continu-

are vor fi specificate principalele aspecte ale acestor etape, urmînd ca în capitolele următoare ele să fie reluate în contextul dezvoltării teoretice și prezentării unor cercetări proprii în aceste domenii.

1.5.1. ALGORITMI DE DETECȚIE ȘI DIAGNOSTICARE A DEFECTĂRILOR

Detecția și diagnosticarea defectărilor — identificarea unei relații structurale τ' pentru tripletul (S, F, Φ) — constituie punctul de pornire al oricărei implementări a toleranței la defectări, atunci cînd aceasta este realizată prin reconfigurarea sistemului. Toate acțiunile de localizare a defectărilor sistemului pot fi considerate ca fiind incluse în algoritmul de diagnosticare a defectărilor — etapă a unei metode de implementare a toleranței la defectări pentru realizarea R a unui sistem dat. Într-un astfel de algoritm se disting următoarele abordări:

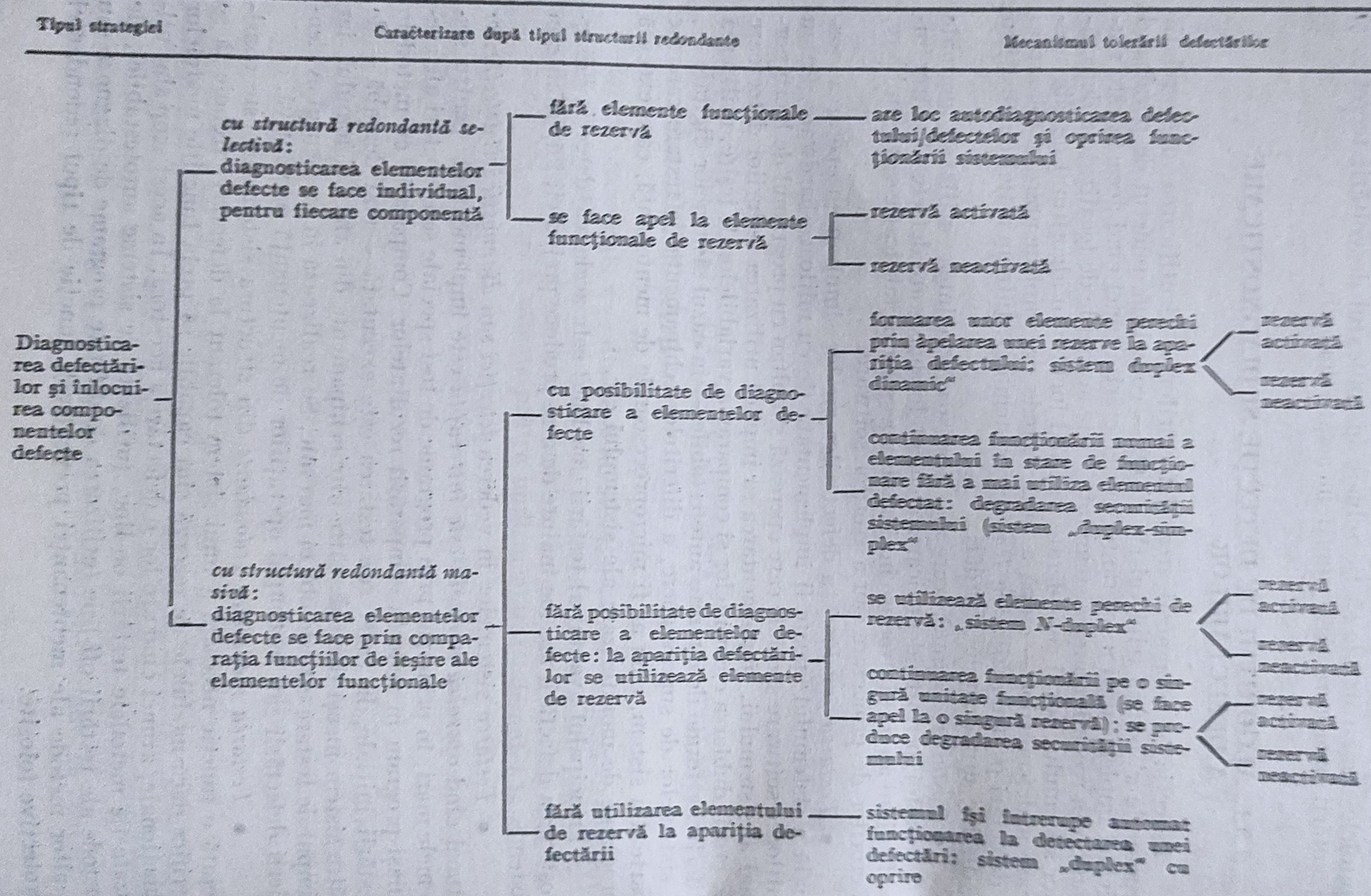
• *Testarea inițială*, care se desfășoară înaintea utilizării normale a sistemului, permite identificarea elementelor hardware defecte ale căror imperfecțiuni au fost introduse în timpul proceselor de fabricație sau asamblare, a erorilor de proiectare sau a erorilor software.

• *Testarea „on line"* a defectărilor are loc simultan cu operarea normală a sistemului; poate fi implementată fie cu mijloace hardware, fie cu mijloace software speciale care operează simultan cu programul de lucru normal al sistemului. Implementarea sa implică utilizarea codurilor detectoare de erori, dublarea elementelor și compararea variabilelor de ieșire, utilizarea diferitelor forme de circuite autotestabile — în cazul sistemelor digitale —, a sistemelor de supraveghere, a diferitelor echipamente de testare care fac parte din sistem, cum ar fi microprocesoarele de mentenanță, care execută programe de monitorizare ale sistemului etc.

Principalul avantaj al testării „on line" este acela că detecția / diagnosticarea defectării are loc înainte de a se produce prejudicii importante în sistem.

• *Testarea* sistemelor în vederea *detecției sau diagnosticării defectărilor* atunci cînd *operarea normală este întreruptă* este implementată pentru sistemele mari, în principal prin programe de test speciale sau repetări ale aceluiași program în vederea comparării rezultatelor. Comparativ cu metodele testării inițiale, în acest caz testarea este executată — mai degrabă — de către sistem însuși decît de către alte echipamente din afara sistemului, iar timpul de testare este de obicei mai mic. Se realizează în momentul evidențierii defectării sau în timpul operațiilor de mentenanță preventivă.

• *Testarea modulelor redondante* din structura sistemului este considerată o funcție necesară a unui sistem tolerant la defectări; trebuie să se verifice dacă modulele de rezervă sînt capabile să preia funcțiile modulelor funcționale, atunci cînd are loc o defectare a acestora. În acest scop sînt utilizate fie metodele testării on-line, (utilizate de sisteme autotestabile), fie metode ale testării off-line (utilizarea de diferite programe de diagnoză sau a altor metode ale mentenanței preventive), funcție de tipul redondanței protective folosite.


Tipul strategiei
Caracterizare după tipul structurii redondante
Mecanismul tolerării defectărilor
Diagnosticarea defectărilor şi înlocuirea componentelor defecte
cu structură redondantă selectivă: diagnosticarea elementelor defecte se face individual, pentru fiecare componentă
fără elemente funcţionale de rezervă
are loc autodiagnosticarea defectului/defectelor şi oprirea funcţionării sistemului
se face apel la elemente funcţionale de rezervă
rezervă activată
rezervă neactivată
cu structură redondantă masivă: diagnosticarea elementelor defecte se face prin comparaţia funcţiilor de ieşire ale elementelor funcţionale
cu posibilitate de diagnosticare a elementelor defecte
formarea unor elemente perechi prin apelarea unei rezerve la apariţia defectului: sistem duplex dinamic"
rezervă activată
rezervă neactivată
continuarea funcţionării numai a elementului în stare de funcţionare fără a mai utiliza elementul defectat: degradarea securităţii sistemului (sistem „duplex-simplex"
fără posibilitate de diagnosticare a elementelor defecte: la apariţia defectărilor se utilizează elemente de rezervă
se utilizează elemente perechi de rezervă: „sistem N-duplex"
rezervă activată
rezervă neactivată
continuarea funcţionării pe o singură unitate funcţională (se face apel la o singură rezervă); se produce degradarea securităţii sistemului
rezervă activată
rezervă neactivată
fără utilizarea elementului de rezervă la apariţia defectării
sistemul îşi întrerupe automat funcţionarea la detectarea unei defectări: sistem „duplex" cu oprire

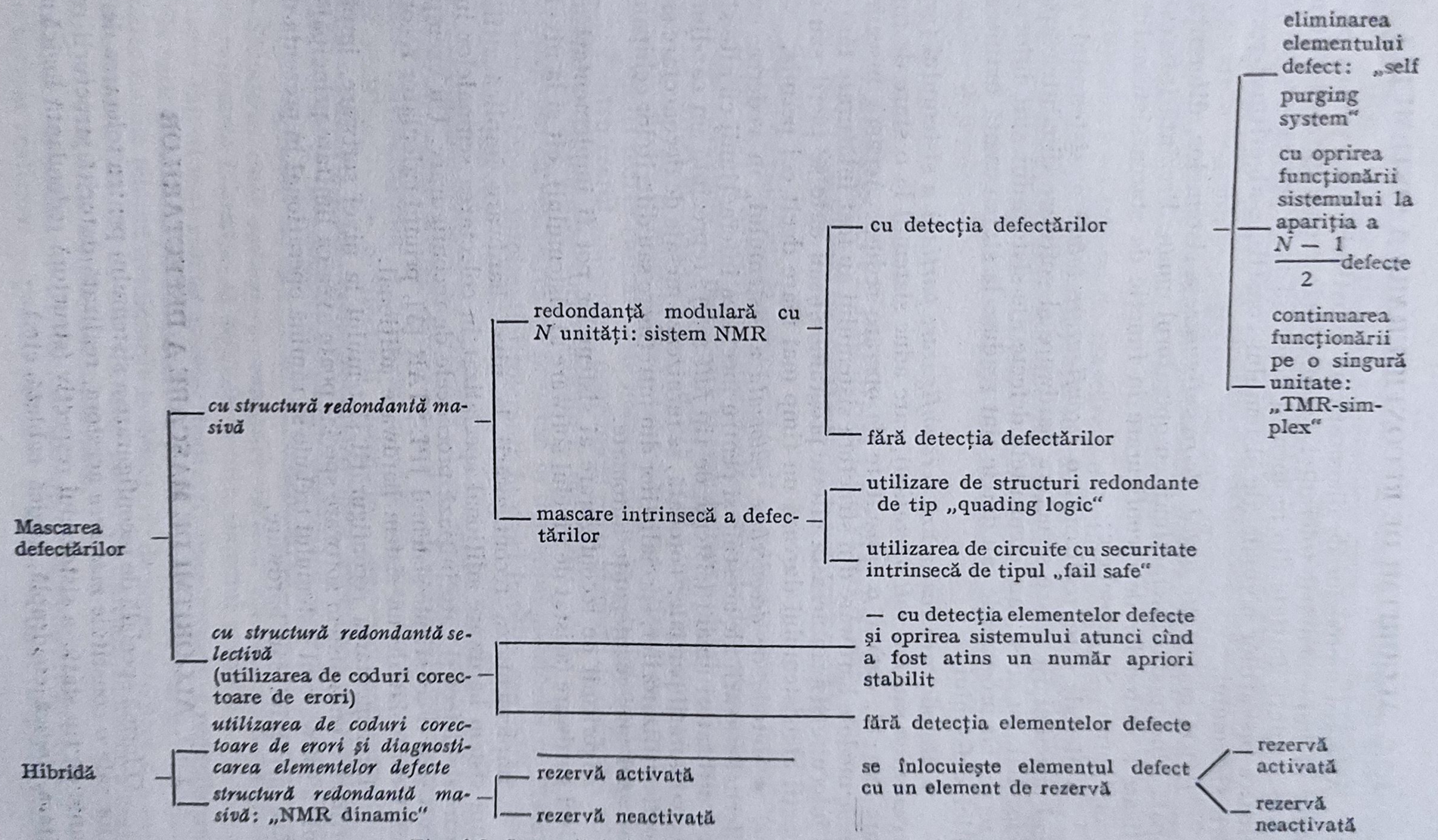


Fig. 1.3. Strategii de implementare a toleranței la defectări.

1.5.2. ALGORITMI DE RECONFIGURARE A SISTEMULUI

Alegerea metodelor de diagnosticare constituie baza pentru pasul următor al implementării toleranței la defectări: reconfigurarea sistemului. Toate acțiunile inițiate din momentul detecției unui semnal-eroare pînă la reluarea operațiilor normale ale sistemului constituie algoritmul reconfigurării sistemului.

Există mai multe *metode de reconfigurare* a sistemelor, diferența principală fiind modul interacțiunii cu operatorul uman. Prezintă interes o clasificare a metodelor de reconfigurare în funcție de starea sistemului după reconfigurare. Se deosebesc:

- Metode care realizează o *reconfigurare totală* a sistemului, atunci cînd sistemul își reface structura hardware și software dinaintea apariției defectării. În acest caz, elementele defecte ale sistemului sînt înlocuite de rezerve, iar programele sau datele sînt readuse la starea avută înaintea apariției defectului.

- Metode care realizează o *reconfigurare parțială* a sistemului (*graceful degradation* sau *fail soft operation*), care aduc sistemul la o stare de funcționare corectă, dar cu o capacitate de operare redusă. Aceasta înseamnă că unele module hardware din structura sistemului au fost înlăturate fără a mai fi înlocuite și prin aceasta cîteva programe și/sau date se pierd, sau cîteva funcții ale sistemului durează un timp mai mare decît cel permis.

- Metode de *deconectare automată* a sistemului, în vederea evitării diferitelor avarii ale acestuia și pentru încetarea interacțiunii cu alte sisteme sau utilizatori umani (sisteme de tip *fail safe*); reprezintă un caz-limită al reconfigurării parțiale. Totodată, se furnizează mesaje de deconectare și diagnostic utilizatorilor, specialiștilor din întreținere sau diferitelor echipamente, în cazul unei mentenanțe automate.

Algoritmii de reconfigurare ai sistemelor pot fi implementați atît la nivel hardware (asistat de nivelul software al sistemului), cît și la nivel software.

Implementarea reconfigurării la nivel hardware implică utilizarea unui sistem hardware adițional specializat în colectarea semnalelor indicatoare de defect și care inițiază procedeele de reconfigurare. Un exemplu în acest sens îl constituie sistemul JPL-STAR [3], primul calculator reconfigurabil controlat de un sistem hardware adițional.

Implementarea reconfigurării sistemului la nivel software, implicînd utilizarea unui sistem software special poate avea ca limitare principală faptul că software-ul sistemului trebuie să rămînă operațional în prezența defectării nivelului său hardware.

1.5.3. ALGORITMI DE MASCARE A DEFECTĂRILOR

O formă specială de reconfigurare a sistemului pentru tolerarea defectărilor sale o constituie *mascarea* acestora, realizată datorită structurii redondante de tip static a sistemului respectiv (structură redondantă logică majoritară, logică cuadruplă, logică cablată etc.).

În acest caz, simptomele defectelor prezente în modulele sistemului nu apar la ieşirile acestuia atît timp cît modulele-rezervă nu sînt afectate — în totalitate sau în majoritate — de un defect, în caz contrar rezultînd defectarea subsistemului sau sistemului respectiv.

Privind din afară un sistem care realizează mascarea defectărilor sale, nu se poate distinge un proces de detecţie a defectărilor urmat de procesul de reconfigurare a sistemului respectiv, din care cauză tehnicile de mascare a defectărilor sînt numite tehnici de redondanţă statică. Elementele de rezervă sînt permanent conectate, mascarea defectărilor realizîndu-se astfel, instantaneu şi automat.

Utilizarea unei tehnici de mascare a defectărilor în implementarea toleranţei la defectări este bazată pe presupunerea că defectările modulelor de rezervă sînt evenimente independente. Din această cauză utilizarea tehnicilor de mascare a defectărilor este dificil de justificat pentru o structură redondantă internă a unui circuit integrat la care este foarte probabilă apariţia defectărilor dependente.

1.6. PERSPECTIVE PRIVIND DEZVOLTAREA SISTEMELOR TOLERANTE LA DEFECTĂRI

Conceptul de toleranţă la defectări (*fault tolerance*), introdus în anii '60, dezvoltat după 1971, a devenit familiar printre cerinţele de performanţă — în special — ale sistemelor de prelucrare şi transmitere a datelor. O analiză a utilizării toleranţei la defectări [3], [6], [8], [22], [36] evidenţiază că doar sistemele pentru care se pune în mod acut problema realizării unei fiabilităţi şi disponibilităţi foarte ridicate este necesar să fie în totalitate tolerante la defectări. Pentru restul sistemelor, doar unele părţi afectate de defectări critice trebuie să fie cu structuri tolerante la defectări.

Considerăm că realizarea de progrese în domeniul sistemelor de prelucrare şi transmitere a informaţiei tolerante la erori implică cercetări avînd următoarele obiective:

• *Acceptarea toleranţei la defectări ca un atribut arhitectural, accesibil şi necesar al sistemelor.* Astfel, devine necesar ca eforturile privind creşterea fiabilităţii calculatoarelor — de exemplu — să nu mai fie fragmentate în domenii restrînse — ale arhitecturii sistemelor, ingineriei software-ului, testării şi verificării proiectării, dezvoltării unor tehnologii sofisticate de asamblare —, ci să fie concentrate asupra dezvoltării unor tehnici globale eficiente de implementare a toleranţei la defectări.

• *Stabilirea unor indicatori cantitativi care să evidenţieze beneficiile aduse de utilizarea toleranţei la defectări* pentru sistemul respectiv.

Costul iniţial mai ridicat necesar realizării unor structuri redondante este acela care determină pe utilizatori să prefere pentru realizarea unor sisteme foarte fiabile tehnici netolerante la defectări; de aceea este necesară introducerea unor indicatori care să justifice investiţia iniţială financiară mai mare prin evidenţierea beneficiilor ulterioare.

Un astfel de indicator trebuie, de exemplu, să evidenţieze reducerea costului necesar realizării — pentru sistemul analizat — unui timp dat ridi-

cat de bună funcționare prin utilizarea toleranței la defectări, comparativ cu alte tehnici de realizare a acestui timp de bună funcționare pentru sistemul intolerant la defectări.

• *Elaborarea unor specificații pentru evaluarea toleranței la defectări.* Definirea indicatorilor de fiabilitate uzuali (funcția de fiabilitate, MTBF etc.) în cazul sistemelor tolerante la defectări la fel ca pentru sistemele netolerante la defectări nu poate evidenția convenabil starea și performanțele sistemului; este de asemenea necesară definirea unor indicatori specifici care să evidențieze caracteristicile procesului de reconfigurare a sistemului la apariția unei defectări.

• *Elaborarea unor modele matematice corespunzătoare care să permită descrierea abstractă a sistemelor din punctul de vedere al implementării toleranței la defectări.*

Astfel, dacă ne referim la sistemele informaționale, acestea pot fi descrise în termenii unei succesiuni de patru sfere (universuri), ordonate în următoarea succesiune: fizică, logică, informațională și externă (cea a utilizatorilor). Fiecare *univers* are o convenție temporală, care determină baza pentru observarea și interpretarea variabilelor care reprezintă structuri informaționale și semnale de control.

Funcționarea corespunzătoare a sistemului, așa cum este ea observată la niveluri externe, poate fi întreruptă de către un „eveniment nedorit"; aceasta își are originea într-unul din universurile interne și poate produce probleme în toate universurile superioare acestuia.

Conceptul de toleranță la defectări a sistemului și metodologiile pentru implementarea sa pot fi explicate în termenii celor patru universuri, a evenimentelor lor nedorite, a algoritmilor de detecție a acestor evenimente și a procedeelor de restabilire a sistemelor.

Pe această bază, este posibilă elaborarea de modele matematice abstracte, deci cu un grad suficient de generalitate, care pot fi apoi particularizate în cazul diferitelor categorii de sisteme, elaborîndu-se pe această bază unele proceduri optimale de proiectare a sistemelor tolerante la defectări.

• *Crearea de sisteme experimentale.* Acestea demonstrează utilizatorilor potențiali performanțele sistemelor tolerante la defectări, demonstrație care nu poate fi egalată de o simulare sau modelare matematică.

Progresele rapide înregistrate în realizarea circuitelor integrate de tip VLSI reliefează perspectiva construirii sistemelor electronice tolerante la defectări la un preț rezonabil. În același timp însă, soluționarea problemelor legate de dezvoltarea circuitelor VLSI conduce la apariția unor noi domenii de cercetare, atît tehnologice, cît și legate de fizica defectelor. Astfel o primă problemă are în vedere dimensiunile reduse ale circuitelor: lățimea conductorilor de conexiune ajungînd de la 10 μm în anii 1970 la 0,1 μm pentru prototipurile realizate azi; aceasta face ca aceste circuite să fie susceptibile la impactul cu particulele α, impacturi nesemnificative pentru cele mai multe circuite de tip LSI. În acest sens devin necesare cercetări viitoare pentru dezvoltarea tehnicilor de verificare, detecție a defectărilor tranzitorii și recuperarea sistemului, în sensul mascării acestor defectări într-o funcționare normală.

A doua problemă este legată de creșterea complexității logice a circuitelor (un „cip" avea circa 1 000 porți logice în anii 1970, 100 000 în 1982 și

pînă la un milion în 1985), ceea ce face ca detecția tuturor erorilor de proiectare să fie puțin probabilă și de aici fiabilitatea potențială mai scăzută a acestor circuite.

Problema erorilor de proiectare nedetectate în circuitele VLSI este similară cu cea a erorilor reziduale în software: în ambele cazuri o proiectare perfectă este foarte costisitoare, iar probele că toate erorile au fost îndepărtate sînt imposibil de obținut.

O soluție pentru cea de-a doua problemă o constituie o *proiectare diversitară* [4], combinată cu tehnicile de toleranță la defectări, cînd N (doi, trei sau mai mulți) specialiști proiectează independent același circuit pornind de la funcțiile de intrare / ieșire ale acestuia. Specificarea acestor funcții trebuie astfel realizată încît să minimizeze probabilitatea în conformitate cu care cele N realizări independente să aibă aceleași erori reziduale. Toate cele N realizări pot conține erori, însă în timpul operării concurente, erorile vor fi detectate dacă simptomele lor nu sînt identice și vor fi corectate dacă vor fi găsite o majoritate de realizări bune la stimulii respectivi. Deci, principalul țel al unei proiectări diversitare — în vederea tolerării erorilor de proiectare — este de a asigura o probabilitate foarte mare că erorile de proiectare reziduale ale sistemului sînt independente, iar simptomele lor nu sînt identice la ieșirea sistemului.

În acest mod se poate rezolva problema erorilor de proiectare ale circuitelor VLSI și a verificării acestor circuite, care, din ce în ce mai complexe, nu mai permit o verificare completă a funcționării lor. Mai mult, mijloacele automate de proiectare și verificare pot fi ele însele supuse unor erori de proiectare.

Totodată, în condițiile creșterii complexității circuitelor integrate, costul generării testelor devine foarte ridicat, ceea ce a făcut să crească interesul pentru proiectarea circuitelor autotestabile.

Odată cu dezvoltarea și extinderea sistemelor de calcul au sporit dificultățile problemelor de rezolvat și de aici cerințele pentru mascarea defectelor — care nu mai pot fi controlate în totalitate —au crescut mai mult ca în trecut.

Referitor la cercetările viitoare privind implementarea toleranței la defectări, se consideră importante următoarele direcții de cercetare:

— tolerarea defectărilor de proiectare atît pentru partea de software a sistemelor, cît și pentru cea de hardware;

— aplicarea tehnicilor inteligenței artificiale în vederea implementării toleranței la defectări;

— testarea circuitelor VLSI — cu un număr limitat de conexiuni externe;

— modelarea sistemelor în mod global, incluzînd software-ul și hardware-ul acestora pentru evaluarea performanțelor toleranței la defectări.

Diferite studii de prognoză prevăd că sistemele de calcul ale anilor '90 vor avea implementată toleranța unui număr mare de defecte fizice sau de-fecte de proiectare, ceea ce subliniază importanța și oportunitatea unor cercetări proprii în acest domeniu.

BIBLIOGRAFIE

1. AHEARN, G.R.; DISHON, Y.; SNIEVELY, R.N., *Design Inovations of the IBM 3830 and 2835 Storage Control Units*, IBM J. Res. Develop., *1*, 1972.
2. ANDERSON, T.; LEE, P.A., *Fault Tolerance*, Prentice Hall International, New Jersey, 1981.
3. AVIŽIENIS, A.; GILLEY, C.G.; MATHUR, F.P.; RENNEL, D.A.; ROHR, J.A.; RODIN, D.K., *The STAR Computer : An Investigation of the Theory and Practice of Fault Tolerant Computer Design*, IEEE Trans. on Computers, C-20, *11*, 1971.
4. AVIŽIENIS, A., *Design Diversity — The Challenge of the Eighties*, Digest of papers, FTCS-12, Santa Monica, 1982.
5. AVIŽIENIS, A., *Fault Tolerant Systems*, IEEE Trans. on Computers, C-25", *12*, 1976.
6. BACIVAROF, ANGELICA-BEATRICE, *Sisteme digitale tolerante la defectări*, Teză de doctorat, Institutul Politehnic Bucureşti, 1980.
7. BALLARD, D.R., *Designing Fail — Safe Microprocessor Systems*, Electronics, *1*, 1979.
8. BRENNAN, J.F., *The Self Repairing Computer*, Research Reports, 5, *1*, IBM, 1969.
9. CARTER, W.C.; BOURICIUS, W.G., *A Survey of Fault Tolerant Architecture and Its Evaluation*, Computer, *1*, 1971.
10. CĂTUNEANU, V.M.; BACIVAROF, I.C., *Fiabilitatea sistemelor de telecomunicaţii*, Ed. Militară, Bucureşti, 1985.
11. CĂTUNEANU, V.M.; MIHALACHE, A. *Bazele teoretice ale fiabilităţii*, Ed. Academiei R.S. România, Bucureşti, 1983.
12. CHANG, H.Y.; SMITH, G.W.; WALFORD, R.B., LAMP: *System Description*, Bell Syst. Tech. Journal, 53, *8*, 1974, p. 141—1449.
13. CHANG, H.Y.; MANNING, E.G.; METZE, G., *Fault Diagnosis of Digital Systems*, W.Y., New York, 1971.
14. COSTES, A.; LANDRAULT, C.; LAPRIE, J.C., *Reliability and Availability Models for Maintained Systems Featuring Hardware Failures and Design Faults*, IEEE Trans. on Computers, C-27, 1978, p. 548—560.
15. FREEMAN, H.; METZE, G., *Fault Tolerant Computers Using „Dotted Logic" Redundancy Techniques*, IEEE Trans. on Computers, C-21, *8*, 1972.
16. GEFFROY, J.C.; COURVOISTER, M., *On-Line Testing and Security of a Distributed System for Process Control*, CompCon, Washington, U.S.A., 1976.
17. HAMMING, R.W., *Error Detecting and Error Correcting*, Bell. Syst. Tech. Journal, 26, 1950.
18. KIME, C.R., *Fault-Tolerant Computing. An Introduction and a Perspective*, IEEE Trans. on Computers, C-24, *5*, 1975.
19. NELSON, G.W., *OPTS-600 On-Line Peripheral Test System*, Proc. AFIPS, FFCC, 1968, p. 45—50.
20. von NEUMAN, J., *Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components*, Annals of Mathematical Studies, 1956.
21. NG, Y, W.; AVIŽIENIS, A., *ARIES-An Automated Reliability Estimation System*, Proc. 1977 Annual Reliability and Maintainability Symp., Philadelphia, 1977, p. 108—118.
22. PILAUD, E., *Conception et valialation des sistèmes informatiques à haute sûreté de fonctionnement*, INP de Grenoble, 1984.
23. RAULT, J.C., *Bibliographie sur la détection et la localisation des défauts dans les circuits logique*, Rapport Thompson CSF, 1975.
24. RENNELS, D.A., *Architectures for Fault Tolerant Spacecraft Computers*, Proc. of the IEEE, 66, 1978, p. 1255—1268.
25. ROTH, J.P., *Hardware Verification*, IEEE Trans, on Comp., C-26, 12, 1977.
26. SHORT, R.A.; GOLDBERG, J., *Soviet Progress in the Design of Fault — Tolerant Digital Machines*, IEEE Trans. on. Computers, C-20, *11*, 1971.
27. SIEWIOREK, D.P., *Multiprocessors : Reliability Modeling and Graceful Degradation*, INFOTECH, London, 1977.
28. STIFFLER, J.J.; HECHT, H., *The Fault Tolerant Space Born Computer*, Digest of papers FTCS-8, Paris, 1978.

29. TAKAOKO, T.; MINE, H., *N-Fail Safe Logical Systems*, IEEE Trans. on Computers, C-20, 5, 1971.
30. TOTH, A.; HOLT, C., *Automated Data Base Driver Digital Testing*, Computer, 1, 1974.
31. TRYON, V.G., *Quadded Logic, in Redundancy Techniques for Computing Systems* 1962, p. 205–228.
32. YANG, S.S.; YANG, S.C., *Multiple Fault Detection for Combinational Logic Circuits*, IEEE Trans. on Computers, C-24, 3, 1975.
33. ZADEH, L.A.; POLAK, E., *Teoria sistemelor*, Ed. Tehnică, Bucureşti, 1975.
34. * * * Colecţia revistei *IEEE Transactions on Reliability*, 1970–1987.
35. * * * *Some features of the Czechoslovak relay computer SAPO*, Nachrichtentechnische Fachberichte, 4, 1956.
36. * * * *Special Issues on Fault Tolerant Computers*, IEEE Trans. on Computers, 1971–1987.

CAPITOLUL 2

STRUCTURI REDONDANTE PENTRU IMPLEMENTAREA TOLERANȚEI LA DEFECTĂRI ÎN SISTEMELE HARDWARE

2.1. CONSIDERAȚII ASUPRA UTILIZĂRII REDONDANȚEI PROTECTIVE ÎN SISTEME

Redondanța — definită ca fiind utilizarea într-un sistem a mai multor elemente decît este necesar pentru îndeplinirea funcțiilor cerute acestuia, astfel încît sistemul să funcționeze satisfăcător chiar în prezența unor defecte — constituie metoda de bază pentru realizarea sistemelor tolerante la defectări. Elementele suplimentare sînt fie elemente hardware, cînd se obține o redondanță structurală de tip hardware, fie elemente software, obținîndu-se o redondanță structurală de tip software sau un timp suplimentar de operare — cînd se obține o redondanță de timp. În acest capitol se vor face referiri la structurile redondante de tip hardware. În figura 2.1 este prezentată o clasificare a diferitelor structuri redondante ce pot fi aplicate sistemelor hardware electronice.

În paragraful următor vor fi prezentate pe larg principalele structuri redondante de tip hardware folosite în vederea protecției semnalelor de ieșire ale sistemului la apariția unui defect.

Lucrările publicate pe plan mondial referitoare la utilizarea redondanței în sisteme se ocupă — în principal — de descrierea diferitelor tipuri de structuri redondante și modelarea matematică a caracteristicilor acestora [12], [19], [36], [42], [43], [58], [63] etc. Analiza acestor lucrări evidențiază următoarele:

(1) Teoria redondanței utilizează cîteva „teoreme-limită" [11], care ghidează proiectarea sistemelor cu structură redondantă; dintre acestea se menționează următoarele:

• Redondanța este o metodă comodă de creștere a fiabilității sistemelor dacă defectările elementelor care compun sistemul sînt evenimente independente.

De altfel, în mod uzual calculele de fiabilitate previzională a sistemelor se bazează pe ipoteza — simplificatoare — a independenței defectărilor elementelor acestora; aceasta se datorește atît facilităților de prelucrare matematică, cît și lipsei unor date de fiabilitate valide privind dependența defectărilor.

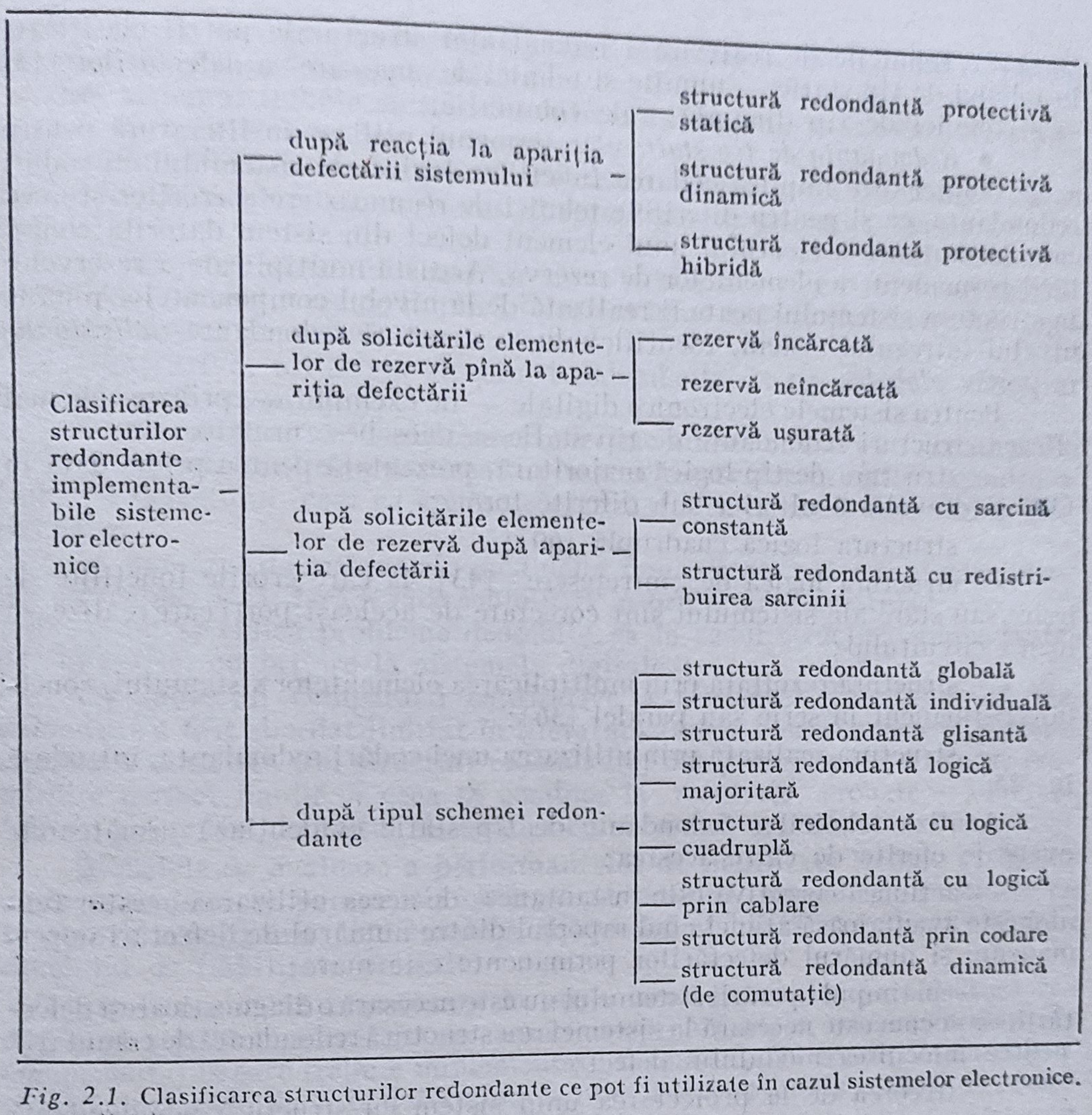


Fig. 2.1. Clasificarea structurilor redondante ce pot fi utilizate în cazul sistemelor electronice.

Practica a demonstrat însă că dimensionarea sistemelor redondante în ipoteza uzuală a independenţei defectărilor poate conduce la evaluări inexacte, obţinîndu-se o supraevaluare a funcţiei de fiabilitate a sistemului. Este demn de menţionat faptul că pentru asigurarea nivelului de fiabilitate preconizat în cazul unor sisteme de comunicaţii operaţionale — pentru care evaluările de fiabilitate în faza de proiectare au fost făcute în ipoteza independenţei defectărilor — a devenit ulterior necesară suplimentarea unora dintre sub-sistemele redondante. De aceea, pentru o proiectare realistă a sistemelor cu structură redondantă este necesar să se ţină seama şi de această dependenţă. O metodă de calcul al fiabilităţii sistemelor redondante luîndu-se în consideraţie efectul defectărilor dependente — defectări de mod comun — a fost dezvoltată în [15].

• Probabilitatea de defectare a unui sistem cu structură redondantă, în termenii unei reţele hardware, scade exponenţial cu creşterea costului alocat sistemului.

(2) Tehnicile de realizare a redondanței structurale pot fi clasificate în tehnici de tip static — numite și tehnici de „mascare" a defectărilor [13] — și tehnici de tip dinamic — de comutație.

• *Redondanța de tip static* este termenul utilizat în literatură pentru acele tehnici care implică codarea funcțiilor logice ale sistemului cu coduri redondante, ca și pentru diferitele tehnici de recunoaștere a erorilor și mascare instantanee a efectului unui element defect din sistem datorită conectării permanente a elementelor de rezervă. Această multiplicare a rezervelor în structura sistemului poate fi realizată de la nivelul componentelor pînă la nivelul întregului sistem, identificîndu-se structuri redondante *individuale*, respectiv *globale*.

Pentru sistemele electronice digitale — de exemplu —, printre cele mai eficace structuri redondante de tip static se deosebesc următoarele:

— structura de tip logică majoritară, prezentată pentru prima dată în [39] și dezvoltată ulterior sub diferite forme;

— structura logică cuadruplă [60];

— structura logică de „întrețesere" [43], la care erorile funcțiilor de ieșire sau stare ale sistemului sînt corectate de aceleași porți care realizează logica circuitului;

— structura rezultată prin multiplicarea elementelor sistemului, conectate permanent în serie sau paralel [36];

— structura realizată prin utilizarea unei codări redondante, introdusă în [25].

Analiza tehnicilor redondante de tip static evidențiază următoarele avantaje oferite de către acestea:

— acțiunea corectivă este instantanee; de aceea utilizarea acestor tehnici este avantajoasă atunci cînd raportul dintre numărul de defectări nepermanente și numărul defectărilor permanente este mare;

— în timpul operării sistemului nu este necesară o diagnosticare a defectărilor așa cum este necesară la sistemele cu structură redondantă de comutație pentru înlocuirea modulului defect;

— trecerea de la proiectarea unui sistem cu structură neredondantă la proiectarea aceluiași sistem, dar cu structură redondantă, este relativ simplă.

Este totuși de reținut că, în cazul sistemelor digitale, structura redondantă de tip static ridică frecvent probleme de *fan-out* și *fan-in*, precum și probleme de complexitate a circuitelor; de asemenea, intervin probleme de sincronizare între modulele de același tip ale sistemului.

• *Redondanța de tip dinamic* (de comutație) este termenul dat tehnicilor care implică utilizarea în sistem a mai multor module cu aceleași funcții, din care doar o parte sînt operante, celelalte fiind în așteptare pentru a fi comutate în momentul identificării unei defectări. Este folosită în realizarea autoreparării sistemelor, atunci cînd comutarea rezervelor se face în mod automat, înlocuindu-se modulul defect cu un modul identic dar în stare de bună funcționare, sau în realizarea reconfigurării, atunci cînd sistemul se reorganizează într-o configurație diferită de cea inițială.

Comparativ cu structura redondantă de tip static, utilizarea pentru implementarea toleranței la defectări a structurii redondante de tip dinamic

implică cîteva cerințe suplimentare pentru sistemul în cauză, care ar putea fi considerate uneori ca dezavantaje:

— sistemul trebuie să tolereze întreruperile și să poată executa o reluare a operațiilor pentru a corecta erorile;

— funcționarea sistemului presupune existența unui comutator sigur care să interconecteze rezervele sau să reconfigureze sistemul, în caz de defectare;

— diagnoza sistemului trebuie să se desfășoare în cursul funcționării normale a acestuia, ceea ce implică folosirea unor metode și tehnici de diagnosticare destul de sofisticate.

Dimpotrivă, utilizarea structurii redondante de tip dinamic conduce la următoarele avantaje:

— este necesar să fie alimentată electric numai o rezervă a sistemului;

— comutatorul care realizează plasarea rezervelor în sistem permite o izolare a defectului, ceea ce apare necesar mai ales în cazul defectărilor dependente;

— numărul de rezerve ale modulelor unui sistem poate fi ajustat pentru o misiune dată, fără a face schimbări în proiectarea acestuia;

— nu se ridică probleme deosebite, ca în cazul structurii redondante de tip static, cu privire la sistemele digitale.

(3) Subiectul comparării cantitative a eficienței diferitelor scheme redondane a fost abordat limitat în literatură: unele lucrări compară performanțele a două sau trei structuri redondante, dar presupun circuitele de restabilire perfect fiabile — ceea ce conduce la rezultate eronate —, iar altele nu țin seama de costul sistemului cu structură redondantă.

Modelele de evaluare a performanțelor de fiabilitate trebuie să indice acele schimbări în parametrii cheie ai sistemului, schimbării rezultate ca urmare a modificării structurale a sistemului stabilită în vederea obținerii nivelului de fiabilitate prescris.

(4) O problemă importantă în proiectarea sistemelor cu structură redondantă constă în identificarea nivelului optim (sistem, subsistem, modul, componentă) la care trebuie implementată redondanța în condițiile anumitor constrîngeri, precum și în determinarea numărului optim de unități de rezervă pentru fiecare tip de structură.

(5) Alte cercetări au în vedere structurile redondante implementate în tehnologiile LSI sau VLSI, precum și dezvoltarea unor modele de evaluare a performanțelor de fiabilitate.

2.2. SCHEME DE IMPLEMENTARE A REDONDANȚEI LA NIVEL HARDWARE

Aplicarea redondanței în sens protectiv la nivel hardware este considerată — în momentul de față — o metodă eficientă pentru realizarea toleranței la defectări și evident pentru îmbunătățirea parametrilor de fiabilitate ai sistemului respectiv.

Implementarea redondanței, în scopul alegerii schemei convenabile, presupune precizarea apriorică a nivelului la care aceasta trebuie aplicată: componentă, modul funcțional sau la nivelul întregului sistem.

2.2.1. STRUCTURI REDONDANTE PROTECTIVE STATICE DE TIP INDIVIDUAL ŞI GLOBAL REZULTATE PRIN MULTIPLICARE

Aceste tipuri de structuri redondante se pot aplica fie la nivelul componentelor sau modulelor funcționale — cînd se realizează o structură redondontă de tip *individual* — , fie la nivelul întregului sistem, obținîndu-se o structură redondantă de tip *global* (*general*).

În figura 2.2 se prezintă modelele logice de fiabilitate pentru structurile redondante rezultate prin multiplicarea componentelor [14]. În figura 2.2*a* este dat modelul logic de fiabilitate al unui sistem neredondant format din n elemente funcționale care intervin în buna funcționare a sistemului după următoarea logică: sistemul funcționează corect cînd funcționează bine elementul *1* și elementul *2* ... și elementul n. Aceasta explică conectarea în serie a celor n elemente — module funcționale — în modelul logic de fiabilitate al sistemului, căci defectarea unuia dintre elemente atrage întreruperea lanțului celor n elemente. Conectarea în paralel din punctul de vedere al fiabilității a modulelor funcționale sau a celor k sisteme în schema 2.2*b* indică logica după care cele $(k - 1)$ rezerve asigură buna funcționare a modulului/ sistemului: pentru buna funcționare a modulului i trebuie să funcționeze bine elementul 1_i sau 2_i sau ... ; altfel exprimat, modulul i se defectează cînd se defectează elementul 1_i și elementul 2_i ... ... și elementul k_i. Aceeași logică este și în cazul structurii redondante globale (fig. 2.2*c*): sistemul redondant este în stare de bună funcționare cînd sînt în stare de bună funcționare sistemul *1* sau sistemul *2* sau ... ; sistemul redondant se defectează sînd se defectează sistemul *1* și sistemul *2* și ... și sistemul k.

Modul de conectare *fizică* a elementelor de rezervă într-o astfel de structură poate să coincidă — sau nu — cu modelul logic de fiabilitate.

Astfel, în figura 2.3 este indicat un exemplu tipic de aplicare a structurii redondante prin multiplicare. Schema 2.3*a* asigură o protecție la defectarea prin scurtcircuit a uneia dintre diode, schema 2.3*b* asigură o protecție la defectarea prin întrerupere a uneia dintre diode, iar schema 2.3*c* asigură protecția la ambele tipuri de defectări: scurtcircuit sau întrerupere.

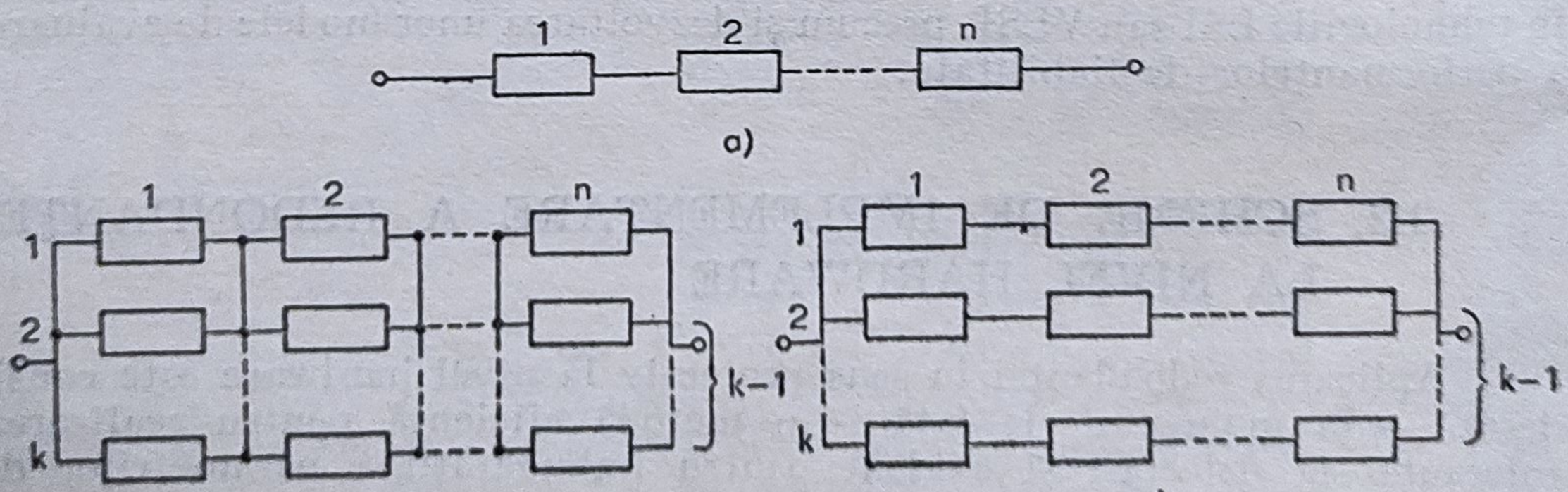


Fig. 2.2. Modele logice de fiabilitate ale unui sistem redondant rezultat prin multiplicarea componentelor:

a — sistem neredondant; *b* — sistem cu structură redondantă individuală; *c* — sistem cu structură redondantă globală.

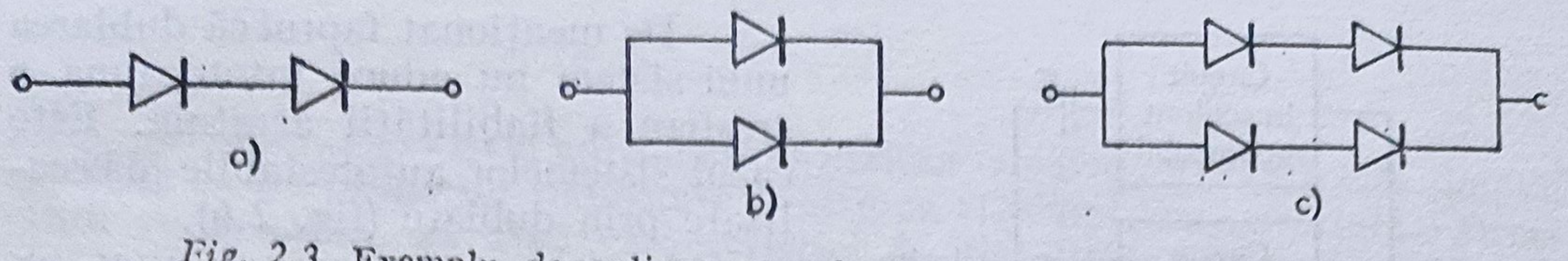


Fig. 2.3. Exemplu de aplicare a redondanței în cazul unei diode:
a — pentru protecția la scurtcircuit; *b* — pentru protecție la întrerupere; *c* — pentru protecție la scurtcircuit și întrerupere.

În figura 2.4 este evidențiată configurația unui sistem de prelucrare a informației cu structură redondantă, rezultată din dublarea sistemului originar. Ambele sisteme ale structurii funcționează „on-line" și execută sarcini identice cu scopul de a compara ieșirile, astfel încît sistemul funcționează chiar dacă unul dintre ele se va defecta.

Se poate introduce — suplimentar — și un sistem de verificare, acesta urmînd să indice care din cele două sisteme de procesare este defect.

Tipurile de structuri redondante individuală și globală sînt aplicabile îndeosebi sistemelor de tip analogic. Utilizarea acestor structuri redondante pentru sistemele digitate în această formă — fără a lua măsuri de sincronizare între module — poate conduce în unele cazuri la sisteme cu caracteristici de fiabilitate necorespunzătoare. De exemplu, aplicîndu-se o astfel de structură unui circuit basculant monostabil (fig. 2.5), se observă că atunci cînd unul dintre circuite se va defecta, astfel încît impulsul de la ieșire va lipsi, schema își va îndeplini funcția ca un întreg, căci vor fi în funcțiune cel de-al doilea, cel de-al treilea circuit monostabil ș.a.m.d.; dacă defectarea unuia dintre circuite este de așa natură încît acesta va genera impulsuri de durată incorectă sau va trece în regim astabil, întreaga schemă se va defecta. În consecință, luîndu-se în considerație acest tip de defectare, performanțele de fiabilitate vor scădea prin aplicarea redondanței și nu vor crește, așa cum era de așteptat. Prin urmare, conectarea modulelor funcționale într-o structură redondantă necesită măsuri de protecție adecvate.

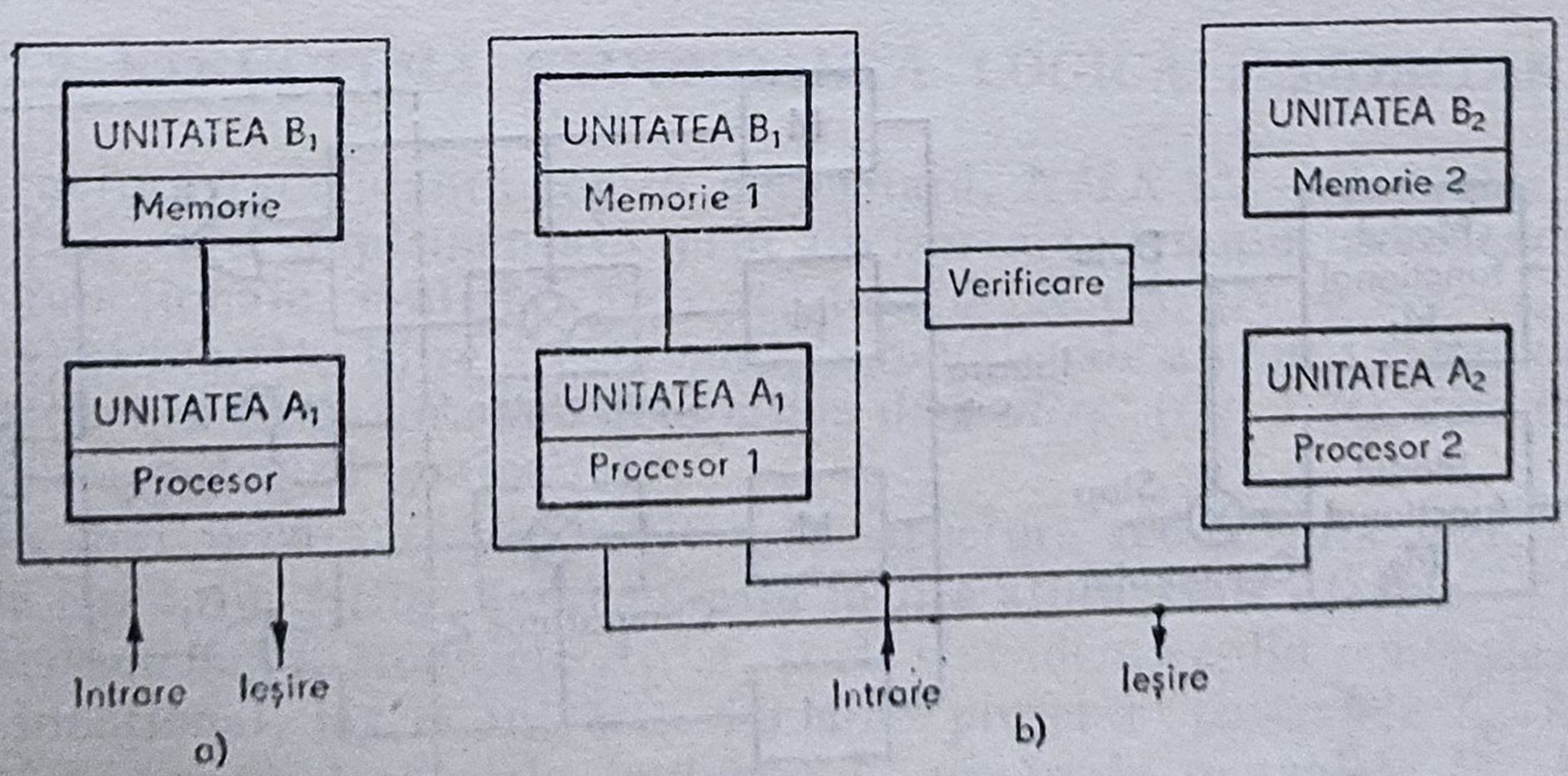


Fig. 2.4. Utilizarea redondanței prin dublare în cazul unui sistem:
a — sistemul originar; *b* — sistemul redondant.

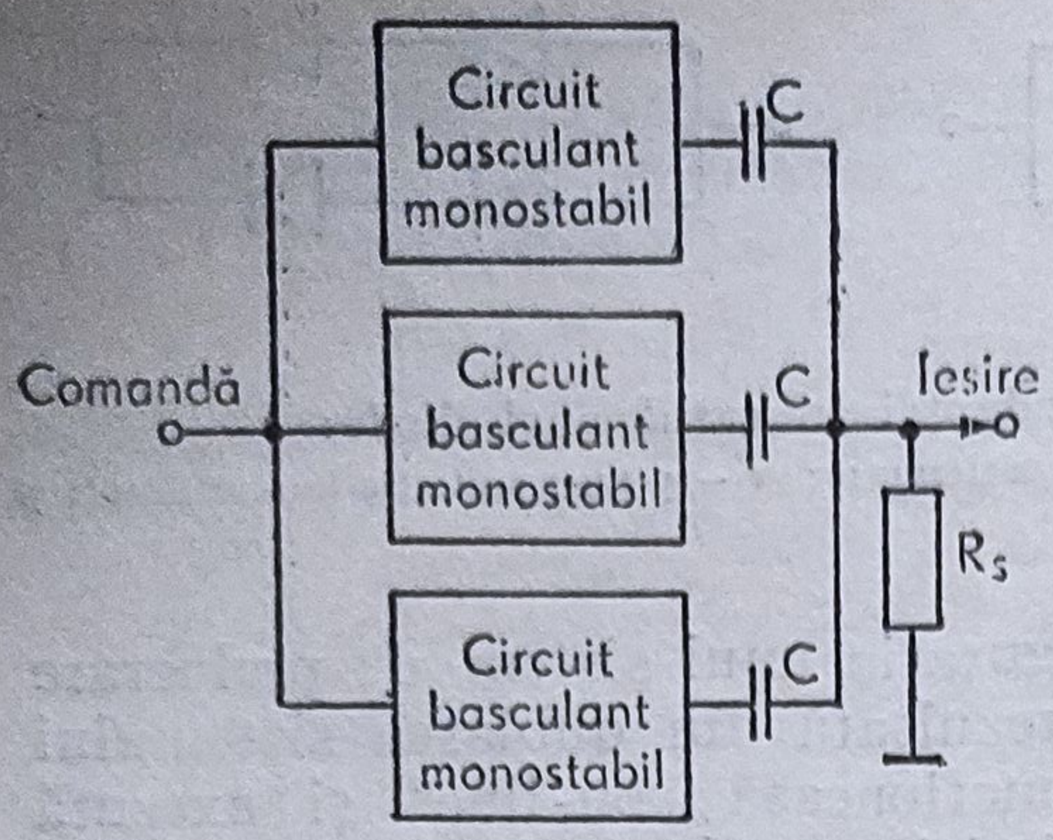


Fig. 2.5. Aplicarea redondanței prin multiplicarea unui circuit basculant monostabil.

De menționat faptul că dublarea unui sistem nu aduce întotdeauna o creștere a fiabilității acestuia. Este cazul sistemelor autotestabile [5] realizate prin dublare (fig. 2.6).

În figura 2.6 *a* este indicată structura unui sistem autotestabil (dezvoltat pe larg în cadrul capitolului 3), format din două module funcționale identice, *M*, și un comparator pentru semnalele de ieșire ale acestora. Comparatorul evidențiază defectarea unuia dintre module prin formarea semnalului corespunzător de „stop". Sistemul cu această structură își îndeplinește funcția — de a evidenția prezența unui defect în sistem — atît timp cît nu intervine același tip de defect *simultan* în cele două module funcționale, *M*.

Pentru a crește performanțele de fiabilitate ale acestui sistem i se poate da o structură redondantă. O soluție posibilă este aceea evidențiată în figura 2.6*b*, rezultată din dublarea sistemului *a* pentru a asigura creșterea disponibilității acestuia. Apariția unei defectări la unul dintre modulele *M* nu mai conduce la formarea semnalului „stop", iar informația falsă va fi blocată spre ieșirea sistemului de către una dintre porțile G_1 corespunzătoare ale multiplicatorului. Semnalul de ieșire va fi dat doar de grupul de module *M* în stare de bună funcționare.

Pentru sistemul analizat, prezența unei defectări va fi evidențiată doar atunci cînd se defectează — în același mod sau în moduri diferite — — două module din cele două grupe. Dacă două unități ale aceluiași grup sînt defecte, iar cele două defectări conduc la același semnal de ieșire — fals — comparatorul nu va putea bloca trecerea acestei informații false spre ieșire.

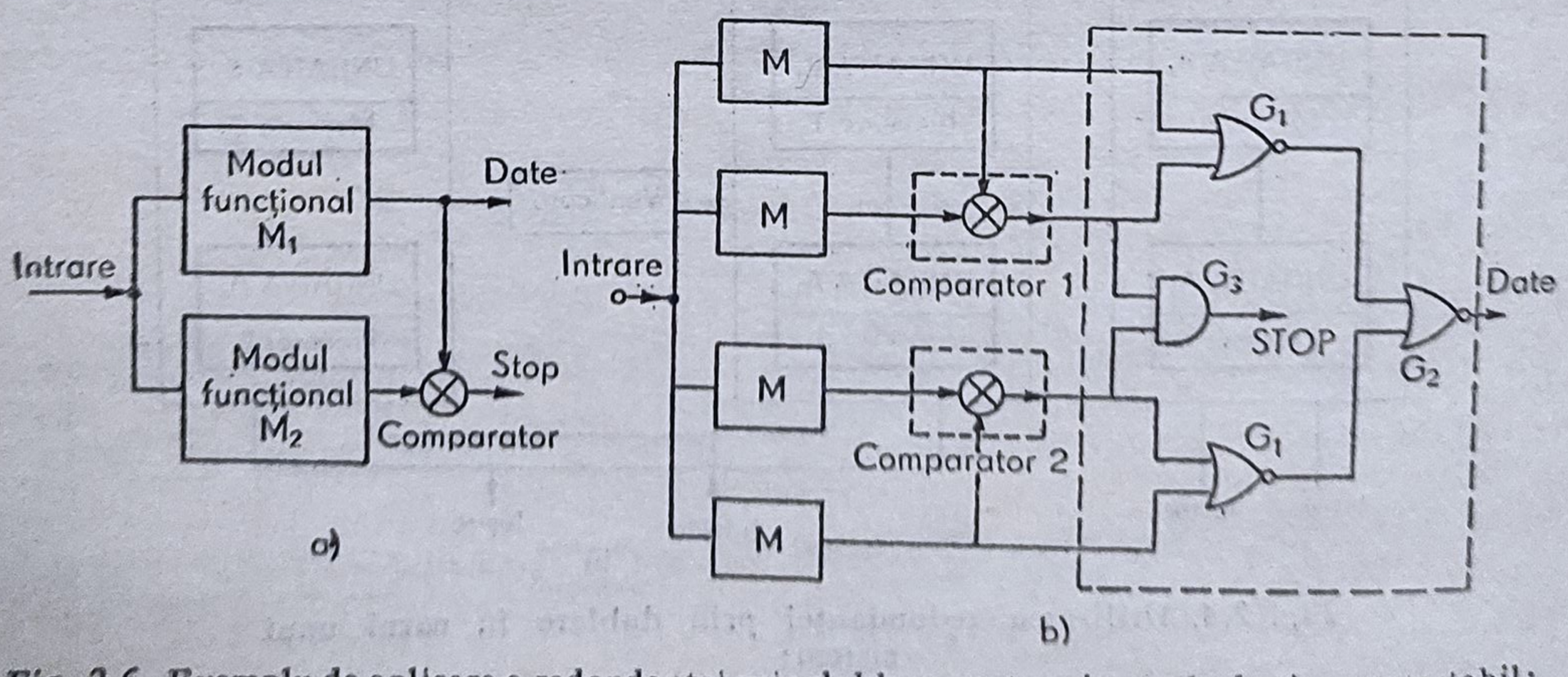


Fig. 2.6. Exemplu de aplicare a redondanței prin dublare pentru sistemele de tip autotestabil:
a — structura unui sistem autotestabil; *b* — sistem autotestabil cu structură redondantă.

Evident, circuitele adiționale introduse, a căror structură este simplă, trebuie să fie de înaltă fiabilitate pentru ca sistemul, care prezintă o structură redondantă, să aibă caracteristici de fiabilitate ridicate.

În condițiile ipotezelor: modulele funcționale sînt nereparabile, circuitele care asigură conectarea modulelor sînt perfecte, defectările elementelor funcționale sînt independente și staționare, se pot scrie expresiile funcțiilor de fiabilitate pentru cele două structuri redondante analizate:

$$R_{SR_I}(t) = \prod_{i=1}^{n} \left[1 - (1 - R_i(t))^{k_i} \right], \tag{2.1}$$

pentru structura redondantă individuală (fig. 2.2*b*) și

$$R_{SR_G}(t) = 1 - \left(1 - \prod_{i=1}^{n} R_i(t) \right)^{k}, \tag{2.2}$$

pentru structura redondantă globală (fig. 2.2*c*).

S-a notat cu $R_i(t)$ funcția de fiabilitate a elementului i din sistemul neredondant și cu $k_i - 1$ — numărul de rezerve utilizate.

Se demonstrează [22] că pentru același număr de rezerve funcția de fiabilitate a structurii redondante individuale are o valoare mai mare decît funcția de fiabilitate a structurii redondante globale, ceea ce se observă ușor și din analiza calitativă a celor două structuri. Pentru structura redondantă globală defectarea unui element oarecare scoate din funcțiune și celelalte $n-1$ elemente, care se află conectate în serie — din punctul de vedere al fiabilității — cu acesta, ceea ce nu se întîmplă pentru structura redondantă individuală. Rezultă de aici că șansele de bună funcționare ale sistemului cu structură redondantă globală sînt mai reduse decît cele ale sistemului cu structură redondantă individuală.

2.2.2. STRUCTURA REDONDANTĂ LOGICĂ MAJORITARĂ

Structura redondantă logică majoritară, *NMR* (*N* — modular redundancy) [2], este implementată prin divizarea sistemului neredondant în module funcționale, multiplicarea acestora de *N* ori, ($N = 2n-1$, n este un întreg) și inserția între aceste module funcționale a unor sisteme de decizie cunoscute în literatură sub denumirea de „*votere*" (fig. 2.7*c*), care funcționează după o logică majoritară.

Cea mai uzuală configurație este structura redondantă logică majoritară de tip 2 din 3 (fig. 2.7*a*) numită în literatură TMR (*T*riple — *M*odular — *R*edundancy). Modulele A_1, A_2 și A_3 sînt identice din punct de vedere fizic și funcțional, iar la intrările lor sînt prezente semnalele I_1, I_2 și I_3 identice. Sistemul de decizie (voterul) V urmărește semnalele E_1, E_2, E_3 aflate în majoritate la intrarea sa. Astfel, dacă $E_1 = E_2 = E_3$, semnalul de ieșire E_0 va fi identic cu acestea; dacă semnalul E_2 — de exemplu — este incorect, deci $E_2 \neq E_1, E_3$, semnalul de ieșire E_0 va lua valoarea majorității, adică valoarea corespunzătoare lui E_1 și E_3. În acest mod structura redondantă TMR maschează defectarea unuia dintre modulele funcționale *A*.

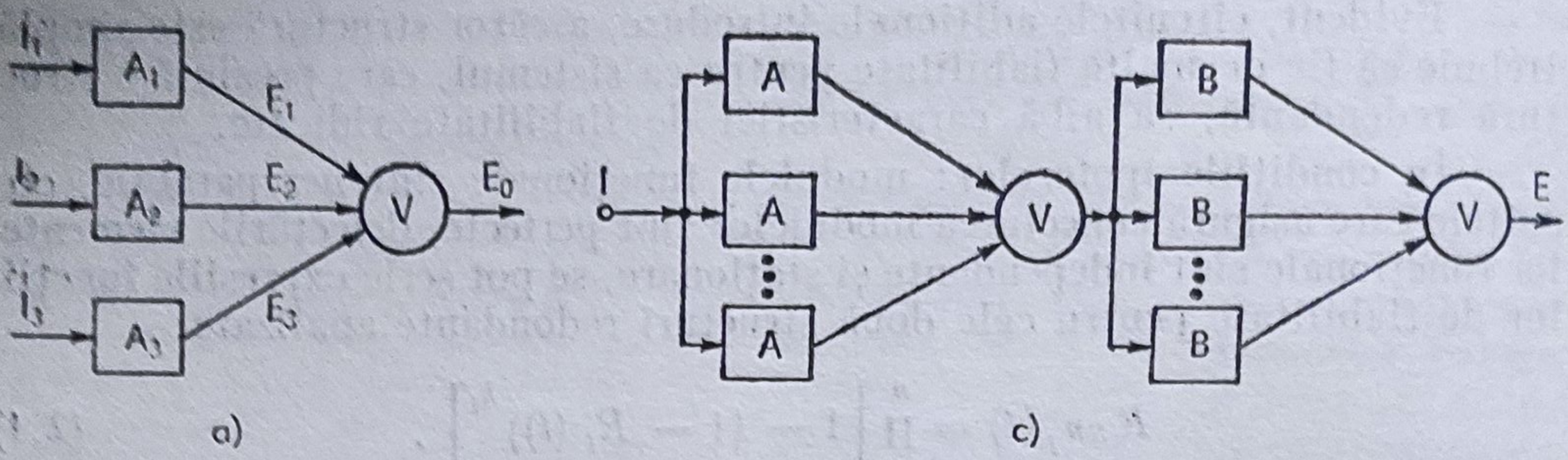


Fig. 2.7. Implementarea structurilor redondante cu logică majoritară:

a — structură redondantă logică majoritară de tip 2 din 3 (TMR); *b* — — sistemul neredondant partiționat în module funcționale; *c* — structură redondantă NMR aplicată sistemului *b*.

Schema TMR nu poate însă corecta două erori de același tip, decît dacă gradul redondanței va crește. În § 2.5 se va arăta că o astfel de schemă poate masca două tipuri de defectări: blocarea în „1" a unui modul și blocarea în „0" a altuia dintre cele trei module funcționale identice.

Deci, structura TMR asigură buna funcționare a sistemului, atunci cînd sînt în stare de bună funcționare toate cele trei module *A* sau oricare grupare de două module, în fiecare caz în parte trebuind să funcționeze corect și voterul *V*.

Generalizîndu-se această observație pentru structura redondantă logică majoritară de tip *n* din $2n - 1$, structura NMR, se precizează că pentru buna funcționare a sistemului trebuie asigurată buna funcționare a oricărei combinații de module cuprinsă între *n* și $2n-1$, celelalte module identice putînd fi defecte; evident, pentru fiecare din aceste combinații este necesară buna funcționare a modulului de decizie. Deci, pentru ca structura analizată să-și atingă scopul de a tolera un număr prestabilit de defectări trebuie utilizat un sistem de decizie cu nivel de fiabilitate înalt.

Cu aceste considerente se poate scrie expresia funcției de fiabilitate (probabilității de bună funcționare) a unui sistem cu structură redondantă logică majoritară de tip *n* din $2n-1$:

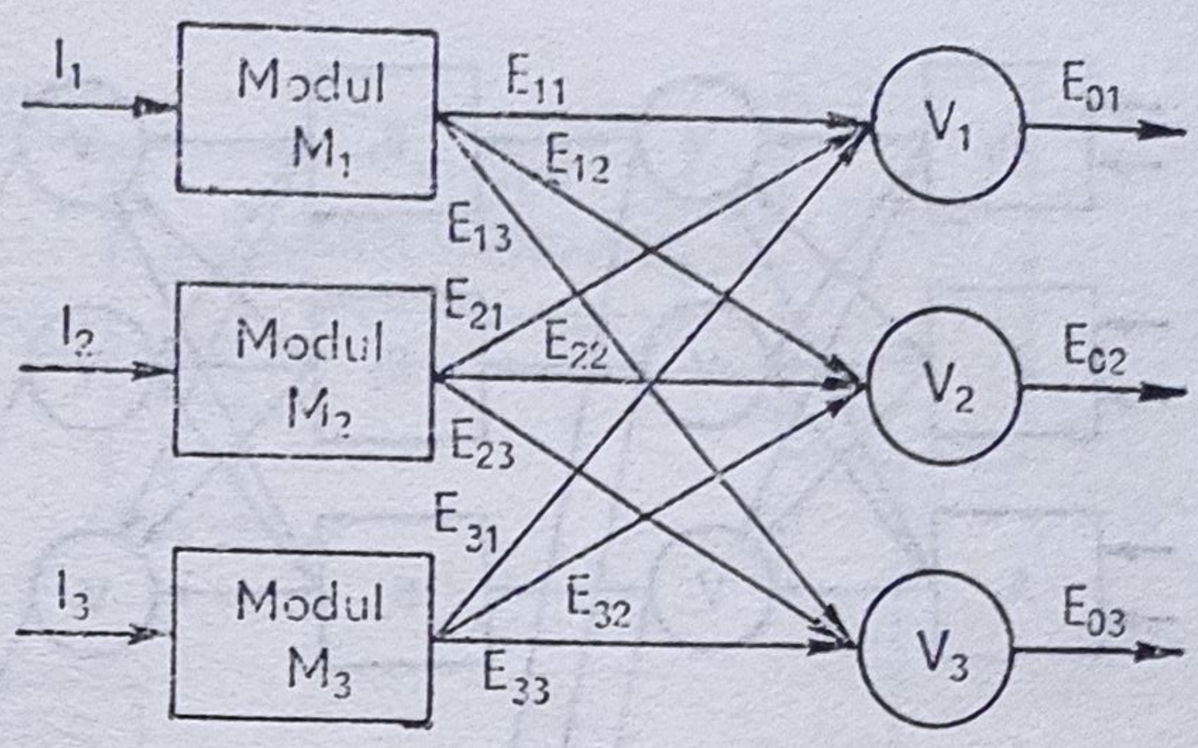
$$R_{SR}(t) = \left[\sum_{j=n}^{2n-1} C_{2n-1}^{j} R(t)^{j} \; (1 - R(t))^{2n-1-j} \right] R_{V}(t), \qquad (2.3)$$

în care $R(t)$ este funcția de fiabilitate a unui modul funcțional, iar $R_V(t)$ — funcția de fiabilitate a voterului.

Se poate introduce o multiplicare a voterelor, rezultînd o structură redondantă logică majoritară în configurație multiplă, cu performanțe de fiabilitate mai bune. În figura 2.8 se prezintă o structură redondantă logică majoritară de tip 2 din 3 în configurație multiplă.

Structura se poate aplica global sistemului sau la nivel de subsisteme/module funcționale. În figura 2.9*a* s-a considerat pentru exemplificare o rețea neredondantă divizată în module funcționale. Modulele sînt triplate și conectate la un sistem de decizie triplat, rezultînd versiunea TMR a rețelei. Normal liniile de intrare și ieșire pentru un modul reprezintă un bus

Fig. 2.8. Structura redondantă logică majoritară de tip 2 din 3, în configurație multiplă.



de date. De aceea, în acest context voterul și modulul funcțional trebuie proiectate ca operatori de vectori și nu ca operatori ai unui singur bit de informație.

O problemă cheie în proiectarea sistemelor cu structură redondantă logică majoritară — și în particular a sistemelor TMR — este dimensiunea modulelor cărora li se aplică redondanța; aceasta poate varia de la un simplu circuit avînd cîteva componente, pînă la un întreg sistem de calcul. Funcție de dimensiunea modulului funcțional structura sistemului de decizie diferă. Problema identificării unui optim al implementării acestei structuri redondante va fi abordată în § 2.5.

O altă cale de creștere a disponibilității sistemelor TMR constă în introducerea unor detectoare de defectare a modulelor, ceea ce permite identificarea modulului defect (fig. 2.10) și apoi înlocuirea acestuia cu un alt modul în stare de bună funcționare. După această înlocuire sistemul TMR este adus la performanțele inițiale. De accea, o astfel de abordare este atractivă pentru sistemele cu cerințe de disponibilitate ridicată și care permit acțiuni de mentenanță, deoarece se obține un timp de indisponibilitate al sistemului foarte redus. Dacă în modulele funcționale poate fi identificată o mulțime de stări interne — uzual, cazul sistemelor de calcul — trebuie luate măsuri de a transfera rezervei, înainte ca aceasta să devină activă, o copie a stărilor curente ale celor două sisteme supraviețuitoare.

Pentru sistemele de calcul reparabile, cu redondanță de tip logică majoritară, pot fi identificate două moduri de partiționare a sistemului în module. O primă abordare constă în utilizarea unei structuri redondante globale, în acest caz un modul fiind reprezentat de întregul calculator; această partiționare se utilizează mai ales atunci cînd nu pot fi identificate defectele în interiorul calculatorului. În acest caz voterul este introdus în sistemul de comunicații între calculatoare.

O altă abordare posibilă presupune partiționarea calculatorului în module mai mici — procesoare, unități de memorie etc. — , unități ce pot fi reparabile individual, caz în care voterul va fi situat pe traiectul magistralelor interne ale sistemului.

De menționat că — datorită componentelor fizice ce intră în structura modulelor funcționale — vor apărea întîrzieri în propagarea informației prin aceste module ale sistemului redondant. De aceea, voterul trebuie astfel proiectat încît să introducă o întîrziere a funcției-răspuns pentru a alinia la intrare toate semnalele de intrare. Această întîrziere poate fi, de exemplu, o perioadă de tact. Logica de decizie trebuie astfel proiectată încît să se

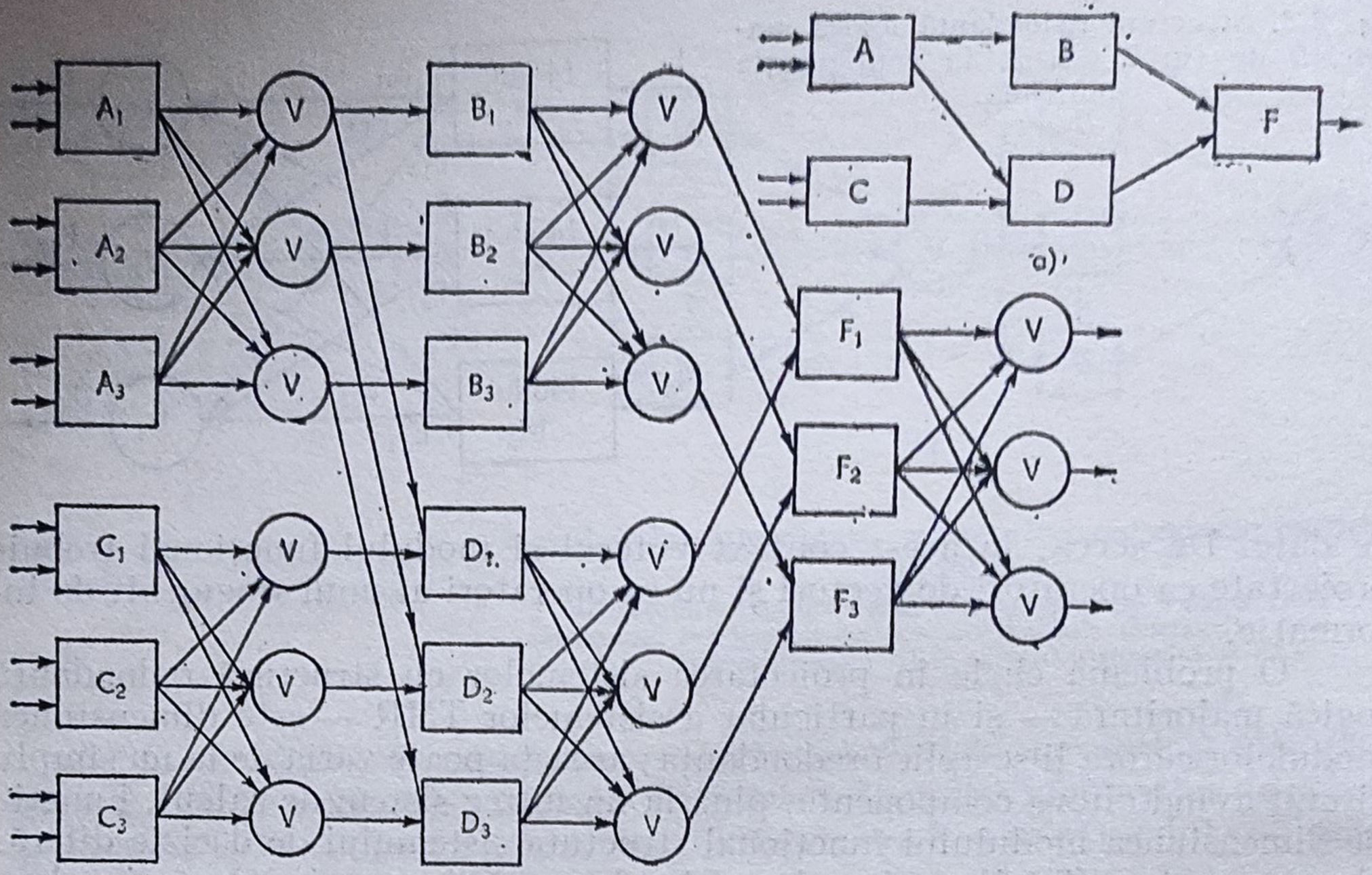


Fig. 2.9. Implementarea structurii redondante logică majoritară de tip 2 din 3, în configurație multiplă:

a — sistemul neredondant partiționat în modele funcționale; *b* — sistemul în structură redondantă.

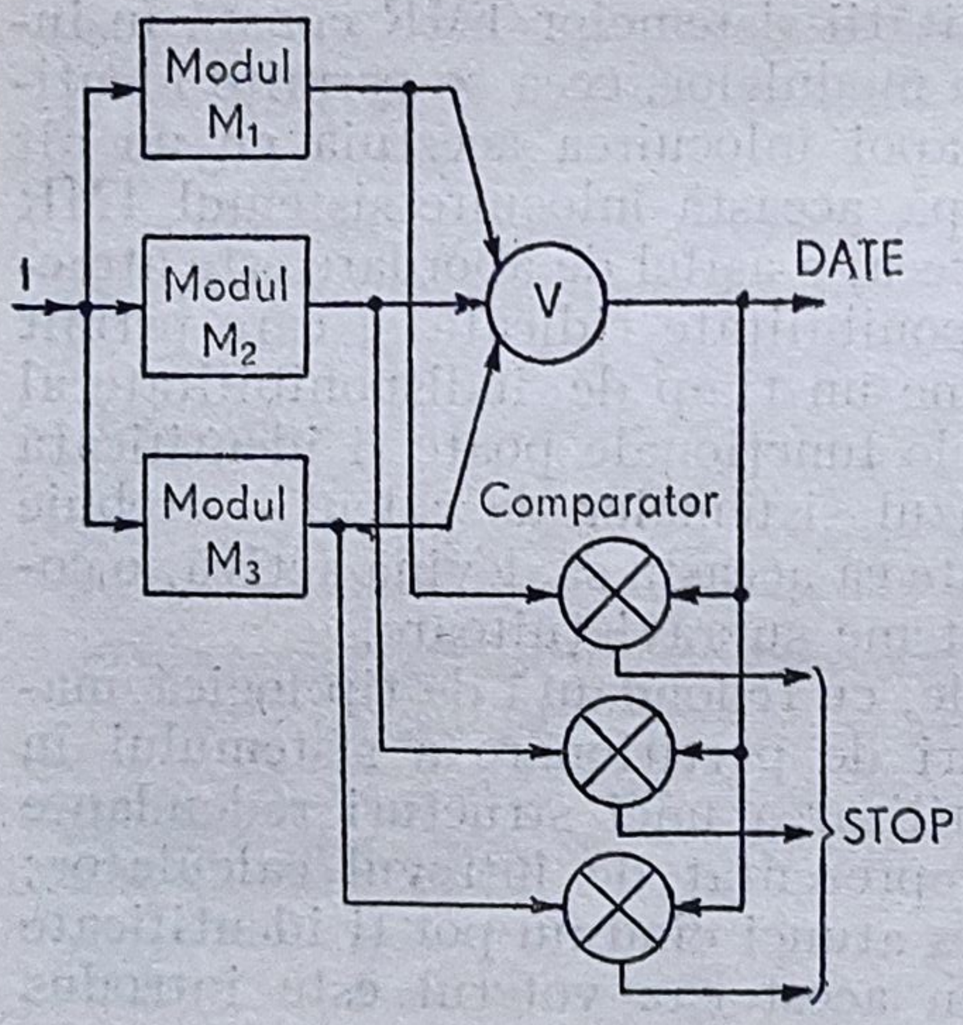


Fig. 2.10. Sistem TMR de tip autotestabil.

întîrzierile fizice introduse de modulele componente ale structurii. Desigur, introducerea unei întîrzieri în funcția-răspuns înrăutățește performanțele sistemului.

Majoritatea sistemelor de decizie sînt implementate hardware. Pentru a se evita întîrzierile diferite introduse de modulele funcționale componente ale structurii se introduce un timp de referință comun. Tipic, este utilizat un generator de tact comun pentru toate elementele structurii, dar aceasta va influența performanțele de fiabilitate, devenind punctul „slab" al sistemului. De aceea, pentru a crește fiabilitatea sistemului pot fi utilizate generatoare de tact redondante [18].

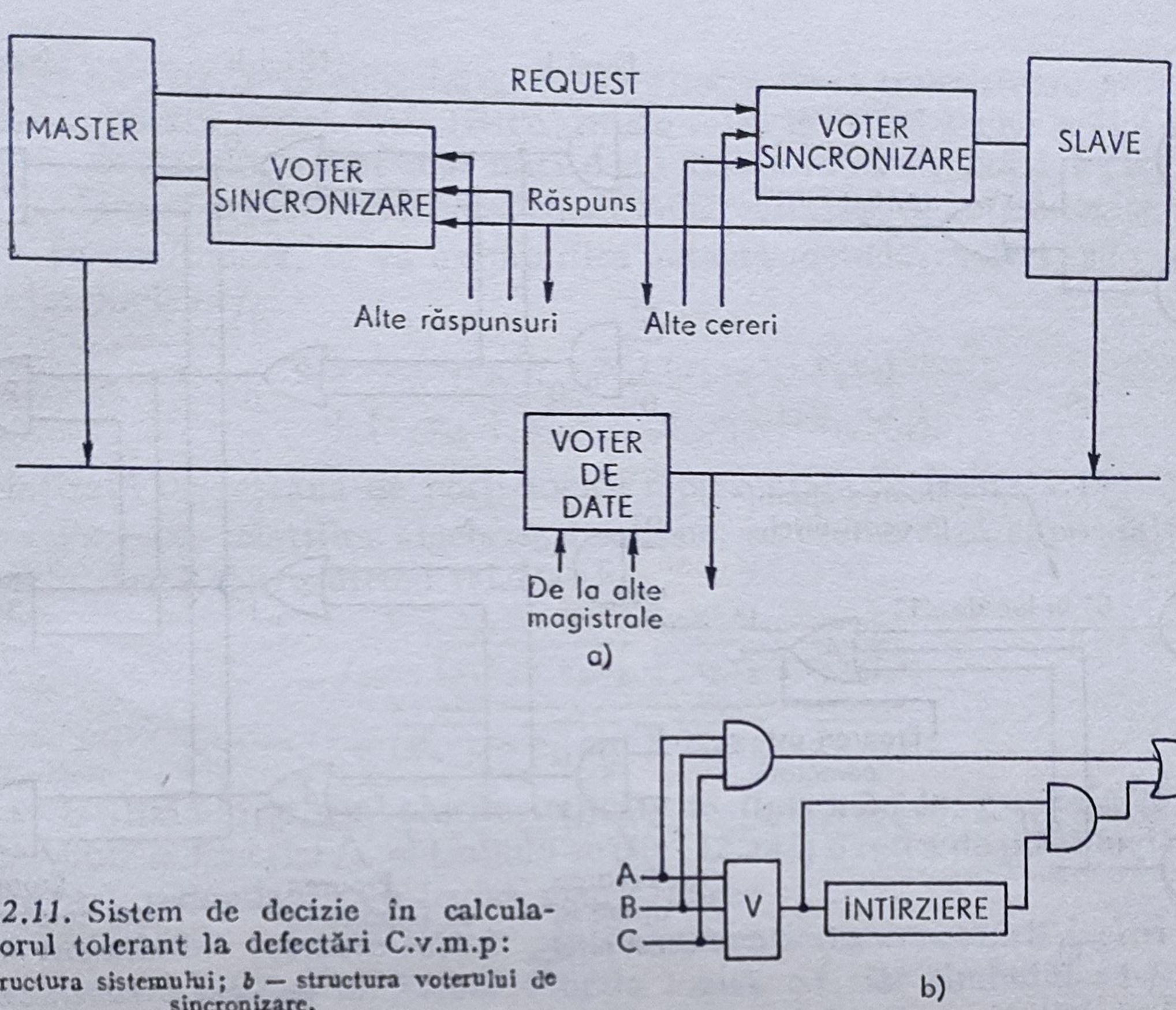


Fig. 2.11. Sistem de decizie în calculatorul tolerant la defectări C.v.m.p:

a — structura sistemului; *b* — structura voterului de sincronizare.

De exemplu, în sistemul de calcul cu structuri tolerante la defectări C.v.m.p. [35] sistemul de decizie este divizat în două părți: voterul pentru sincronizare și voterul de date. Voterul pentru date ia decizie asupra datelor prezente pe trei magistrale de date, iar voterul de sincronizare ia decizie asupra semnalelor de control. Structura voterului de sincronizare este indicată în figura 2.11. Calculatoarele din structură formează semnalul „REQUEST", după care sînt transmise datele pe magistrala de date. Voterul de sincronizare sincronizează funcționarea blocurilor MASTER și SLAVE, astfel încît datele de operare sînt disponibile simultan la voterul de date.

În § 2.4 va fi indicată o soluție de introducere a întîrzierilor, propusă pentru un sistem TMR, în vederea realizării sincronizării.

2.2.3. STRUCTURI REDONDANTE PROTECTIVE STATICE CU LOGICĂ CUADRUPLĂ

Această structură redondantă [60] se aplică sistemelor construite cu circuite logice și constă din multiplicarea de patru ori a circuitelor logice, conectate astfel încît semnalele eronate sînt mixate cu semnalele corecte provenite de la circuitele de rezervă, realizîndu-se în acest fel mascarea unui defect la ieșirea sistemului.

Metoda se bazează pe caracteristicile unor sisteme cu circuite logice de a masca intrinsec unele defectări de blocare în „1" sau „0" ale acestora.

Astfel, analizîndu-se circuitul din figura 2.12*a* se observă că o eroare la ieșirea porții $ȘI_1$ se va propaga la ieșirea porții *SAU* dacă, combinațiile

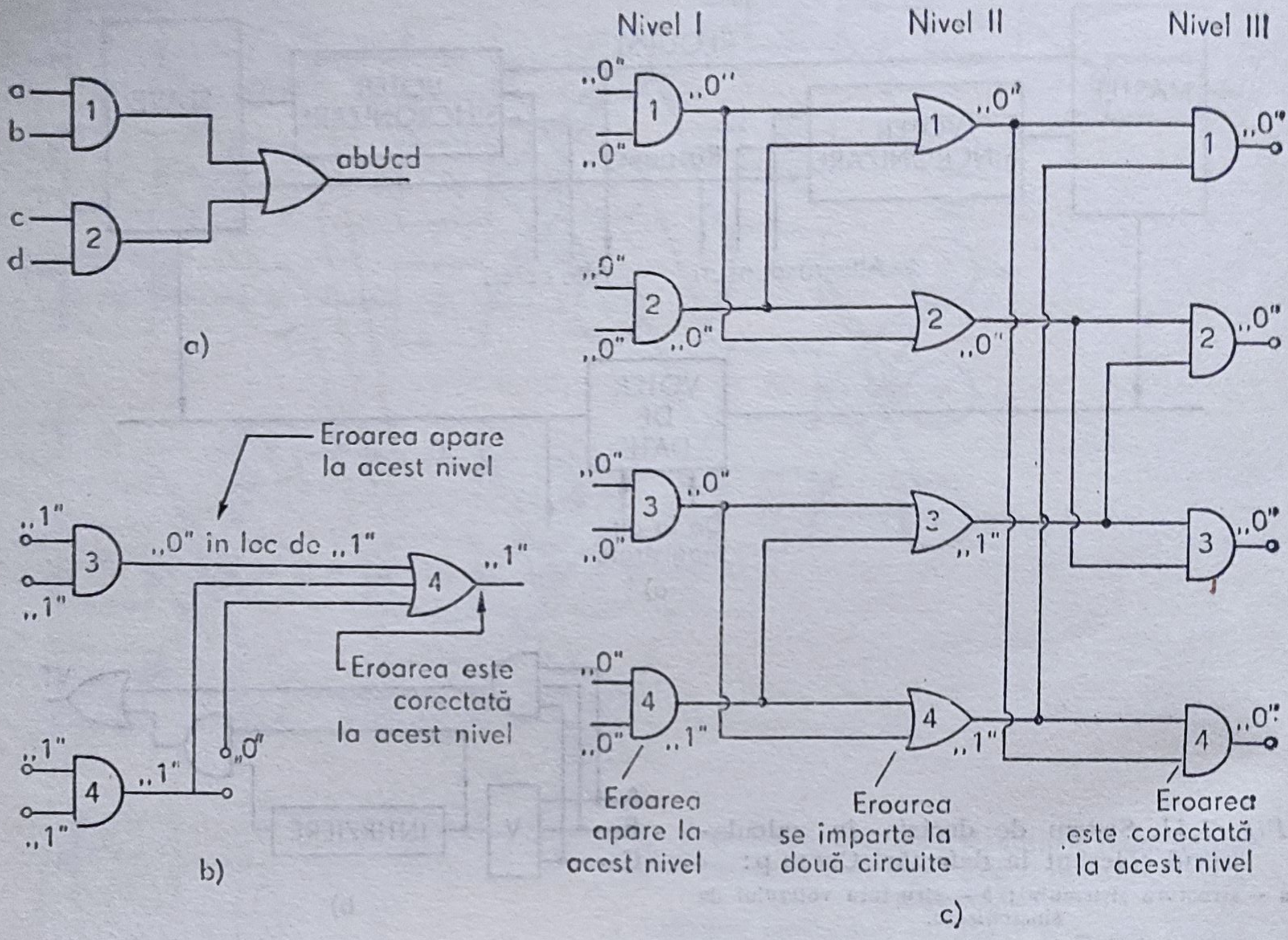


Fig. 2.12. Mascarea erorilor semnalelor logice în circuitele cu logică cuadruplă:
a — circuit logic fără redondanță care maschează unele defectări; *b* — circuit logic redondant; *c* — structură redondantă cu logică cuadruplă.

semnalelor la intrările porților $ȘI_1$, $ȘI_2$ nu sînt identice. În caz contrar, poarta $ȘI_2$ maschează defectarea porții $ȘI_1$, la ieșirea porții SAU semnalul de ieșire apărînd corect. Astfel dacă se dublează porțile $ȘI$ într-o rețea cu astfel de porți logice, după modelul schemei din figura 2.12*a*, va rezulta un sistem care tolerează defectările acestor porți logice (fig. 2.12*b*).

În circuitul indicat în figura 2.12*b* semnalele de ieșire ale porților logice redondante $ȘI_3$, $ȘI_4$ sînt aduse la intrarea porții SAU_4, care va masca eroarea apărută la ieșirea porții $ȘI_3$ — de exemplu.

În figura 2.12*c* se prezintă circuitul desenat în figura 2.12*a* cu structură redondantă logică cuadruplă. Orice eroare apărută la nivelul I este corectată instantaneu de logica circuitului la nivelul II pentru semnale „1" la intrare sau la nivelul III pentru semnale „0" la intrarea porților logice $ȘI_1 \ldots ȘI_4$.

2.2.4. STRUCTURI REDONDANTE PROTECTIVE STATICE CU LOGICĂ PRIN CABLARE

Aceste structuri [19] pot fi de asemenea utilizate pentru rețelele cu porți logice. Tehnica de realizare a acestor structuri constă din conectarea prin cablare a ieșirilor a două porți logice identice, redondante, rezultînd o schemă corectoare de erori. Dacă se conectează colectorii tranzistoarelor *npn* de la ieșirea a două porți logice se formează o poartă logică ȘI (AND);

în mod similar, prin conectarea colectorilor a două tranzistoare *pnp* se formează o poartă logică SAU (OR). Într-o rețea de porți logice astfel cablate șansele de defectare sînt mai mici decît într-o rețea de porți logice fizice, ceea ce face ca această tehnică să conducă la realizarea de rețele logice fiabile.

În continuare, se va exemplifica această metodă. Astfel, se consideră funcția logică, f_2:

$$f_2 = (x_1x_2 + x_3x_4)\,(x_5x_6 + x_7x_8) + \\ + (x_9x_{10} + x_{11}x_{12})\,(x_{13}x_{14} + x_{15}\,x_{16}), \tag{2.4}$$

sintetizată de rețeaua de porți logice reprezentată în figura 2.13.

Pe baza relațiilor algebrice booleene, se prelucrează expresia funcției f_2 dată de (2.4), rezultînd relația:

$$f_2 = \overline{\overline{(x_1x_2} \cdot \overline{x_3x_4} + \overline{x_5x_6} \cdot \overline{x_7x_8)}} + \\ + \overline{\overline{(x_9x_{10}} \cdot \overline{x_{11}x_{12}} + \overline{x_{13}x_{14}} \cdot \overline{x_{15}x_{16})}}. \tag{2.5}$$

Utilizîndu-se simbolurile indicate în figura 2.14*a*, expresia (2.5) permite sinteza funcției f_2, obținîndu-se (fig. 12.14*b*) o rețea de porți logice într-o structură redondantă cu logică prin cablare.

Simbolul (•) este folosit pentru evidențierea conectării ieșirii a două tranzistoare *npn* și realizează funcția logică *ȘI*, iar simbolul (+), pentru evidențierea conectării ieșirilor a două tranzistoare *pnp*, realizînd în acest fel funcția SAU.

Din figura 2.14*b* rezultă că pentru sintetizarea funcției (2.5) au fost necesare 40 porți logice. Pentru realizarea aceleiași funcții, utilizîndu-se o structură redondantă logică cuadruplă sau logică majoritară de tip 2 din 3 ar fi fost necesar un număr de 60, respectiv 45 porți logice, la care se adaugă un număr de porți logice pentru votere.

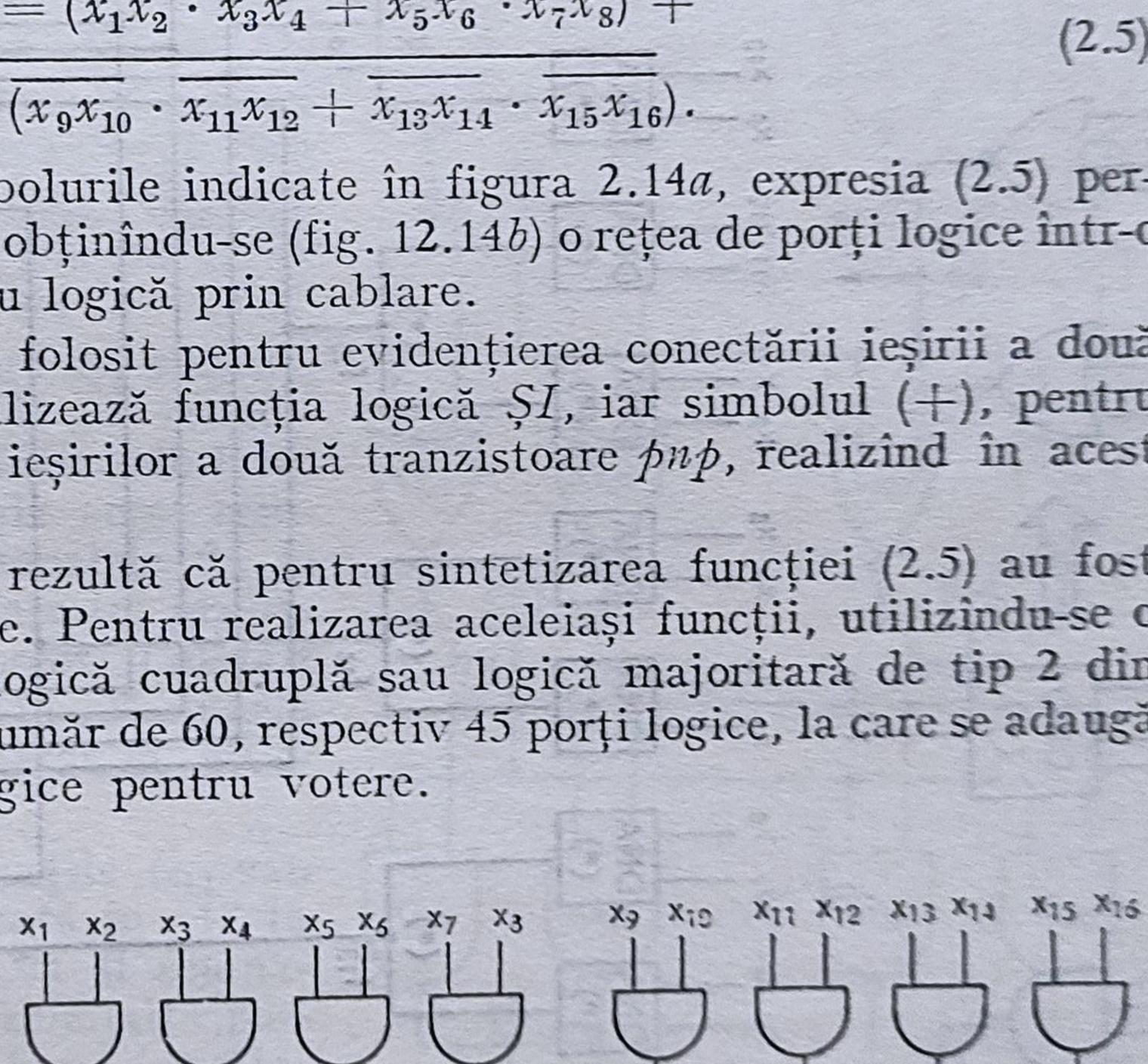


Fig. 2.13. Implementarea funcției f_2 cu o rețea de porți logice.

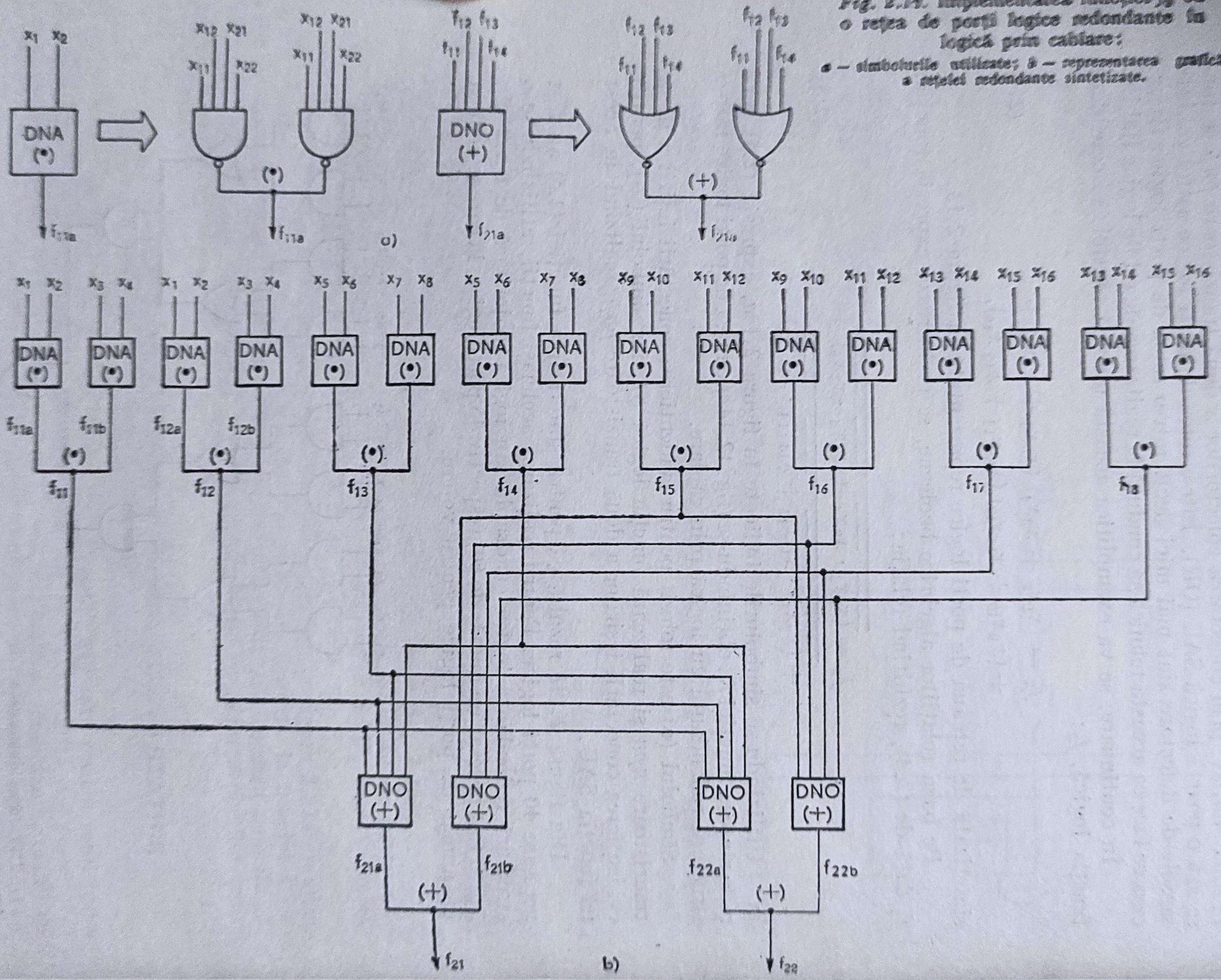


Fig. 2.14. **Implementarea funcției f_0 cu o rețea de porți logice redondante în logică prin cablare:**

a — simbolurile utilizate; b — reprezentarea grafică a rețelei redondante sintetizate.

Mai mult, schema logică prin cablare utilizează numai două interconexiuni redondante între niveluri, pe cînd varianta logică majoritară folosește trei interconexiuni, iar cea logică cuadruplă patru interconexiuni. Al treilea avantaj al structurii redondante cu logică prin cablare constă în viteza de răspuns la semnale a rețelei datorită eliminării unui număr de porți logice fizice și a întîrzierilor lor inerente.

În continuare se va arăta că o astfel de schemă maschează defectările simple ale unei porți logice: intrare blocată în „0" sau „1" logic, respectiv ieșire blocată în „0" sau „1" logic.

Se va urmări pentru exemplificare calea semnalelor redondante x_{11} și x_{12} ale semnalului x_1 (fig. 2.15). Primele două niveluri ale acestei rețele sînt luate din realizarea indicată în figura 2.14*b*. În acest caz primul și al treilea bloc al nivelului 1 (fig. 2.14*b*) corespund porților G_{11}, G_{12}, G_{13}, G_{14} din rețeaua reprezentată în figura 2.15. Se consideră $x_2 = 1$, iar x_3 sau x_4 egale cu 0, pentru ca semnalele f_{11b} și f_{12b} să fie „1". De asemenea, primul și al treilea bloc al nivelului 2 (fig. 2.14*b*) corespund porților G_{21}, G_{22}, G_{23}, G_{24} cu semnalele f_{13} și f_{14} egale cu 0, iar f_{15}, f_{16}, f_{17} sau f_{18} egale cu 1.

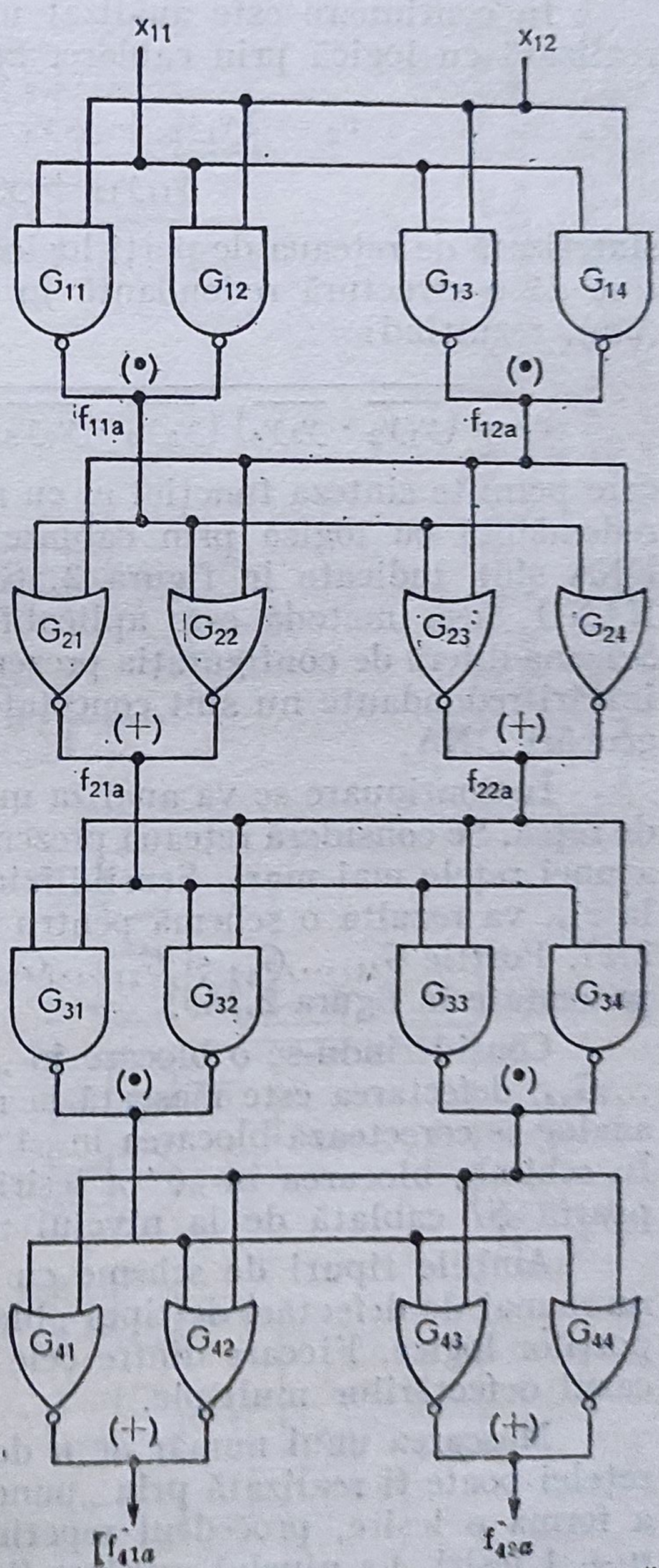


Fig. 2.15. Rețea de porți logice cu structură redondantă cu logică prin cablare.

Dacă linia de intrare x_{11} sau x_{12} a unei porți este blocată în „1" semnalul corect „0" de la intrările celorlalte porți redondante va anula acest semnal „1" incorect, iar semnalele de la nivelul 2 al structurii, datorită grupării redondante, va fi corect. Celălalt mod de defectare-blocare în „0" a uneia dintre intrări — este mascat la nivelul porții *ȘI* cablate.

Dacă una dintre porțile G_{11} sau G_{12} ar avea o ieșire blocată în „1" la ieșirea porții *ȘI* cablate va fi semnalul „0" corect datorită celeilalte porți a grupării în stare de bună funcționare. Blocarea în „0" a ieșirii uneia dintre porțile de la nivelul 1 nu este mascată de poarta *ȘI* cablată, ci la nivelul porților *SAU* cablate.

Deci, la acest nivel apar două tipuri de defectări: mascate la nivelul porților cablate și, respectiv mascate la nivelul 2 datorită redondanței liniilor.

De asemenea, cel de-al doilea nivel, format de porțile $G_{21}, \ldots, G_{24}$,

corectează erori simple. Astfel, o intrare blocată „1" este mascată la nivelul porții SAU cablate corespunzătoare; în mod analog, la nivelul porții SAU cablate este mascată și ieșirea uneia dintre porțile $G_{21},\ldots,G_{24}$ blocate în „0". Însă o intrare blocată în „0" nu mai este corectată la nivelul porții SAU cablate corespunzătoare, ci va fi corectată la nivelul următor datorită mixării cu semnalul redondant f_{22a} (f_{11a}); aceeași situație și pentru blocarea în „1" a ieșirii uneia dintre porțile $G_{21},\ldots,G_{24}$, corecție realizată la nivelul următor.

Deci, o astfel de rețea de porți logice NAND și NOR conectate într-o structură redondantă de acest tip poate masca defectări simple la nivelul uneia dintre porți.

În continuare este analizat un alt exemplu de structură redondantă realizată cu logică prin cablare. Se consideră funcția:

$$v_2 = (y_1y_2 + y_3y_4 + y_5y_6 + y_7y_8)\,(y_9y_{10} + y_{11}y_{12} + y_{13}y_{14} + y_{15}y_{16}), \tag{2.6}$$

sintetizată de rețeaua de porți logice desenată în figura 2.16*a*. Acestei rețele i se dă o structură redondantă în logică cablată. Se prelucrează expresia (2.6), rezultînd:

$$v_2 = \overline{(\overline{y_1y_2}\cdot\overline{y_3y_4})\,(\overline{y_5y_6}\cdot\overline{y_7y_8})}\cdot\overline{(\overline{y_9y_{10}}\cdot\overline{y_{11}y_{12}})\cdot(\overline{y_{13}y_{14}}\cdot\overline{y_{15}y_{16}})} \tag{2.7}$$

care permite sinteza funcției v_2 cu rețeaua de porți NAND într-o structură redondantă cu logică prin cablare ilustrată în figura 2.16*b*. Simbolurile DNA sînt indicate în figura 2.16*c*. Pentru exemplificare s-au ales porți NAND, însă metoda este aplicabilă și în cazul utilizării porților NOR. Schema diferă de configurația prezentată în figura 2.14*b*, în sensul că ambele intrări redondante nu sînt conectate la cele două porți redondante ale unei grupări DNA.

În continuare se va analiza modul de mascare a erorilor într-o astfel de rețea. Se consideră rețeaua prezentată în figura 2.16*b* ca parte componentă a unei rețele mai mari. Sensibilizînd o cale de propagare a erorii de la y_1 la v_{41}, va rezulta o schemă pentru analiză de tipul celei indicate în figura 2.17. Porțile $G_{11},\ldots,G_{14}$ și $G_{21},\ldots,G_{24}$ corespund nivelurilor 1 și 2 ale rețelei prezentate în figura 2.16*b*.

Considerîndu-se o blocare în „0" a intrării uneia dintre porțile $G_{11},\ldots,G_{14}$, defectarea este mascată la nivelul primei porți *ȘI* cablate. În mod analog se corectează blocarea în „1" a ieșirii uneia dintre porțile $G_{11},\ldots,G_{14}$. În schimb, blocarea în „0" a ieșirii uneia dintre porți, este corectată de poarta *ȘI* cablată de la nivelul următor.

Ambele tipuri de scheme cu logică prin cablare protejează circuitul nu numai de defectări de tipul „blocare", dar și de defectări tranzitorii ale porților logice. Fiecare dintre cele două metode poate fi extinsă și pentru cazul defectărilor multiple.

Mascarea unui număr de n defectări care apar într-o zonă limitată a rețelei poate fi realizată prin „punctarea" a $n+1$ porți redondante pentru a forma o ieșire, procedeul repetîndu-se iterativ astfel încît să se obțină $n+1$ ieșiri. La nivelul următor fiecare din cele $n+1$ ieșiri se conectează corespunzător la fiecare grup de porți redondante.

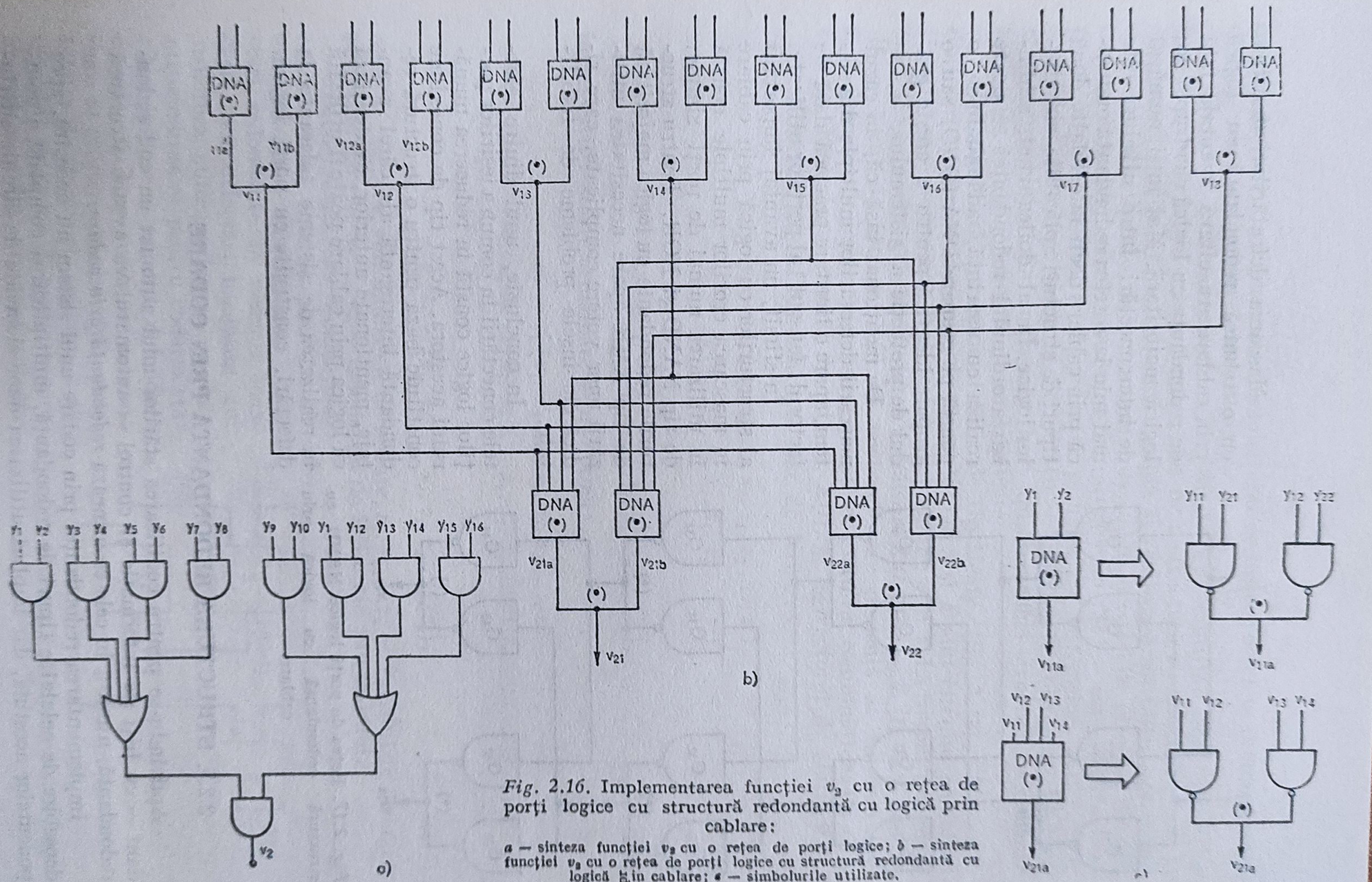


Fig. 2.16. Implementarea funcției v_2 cu o rețea de porți logice cu structură redondantă cu logică prin cablare:

a — sinteza funcției v_2 cu o rețea de porți logice; *b* — sinteza funcției v_2 cu o rețea de porți logice cu structură redondantă cu logică în cablare; *c* — simbolurile utilizate.

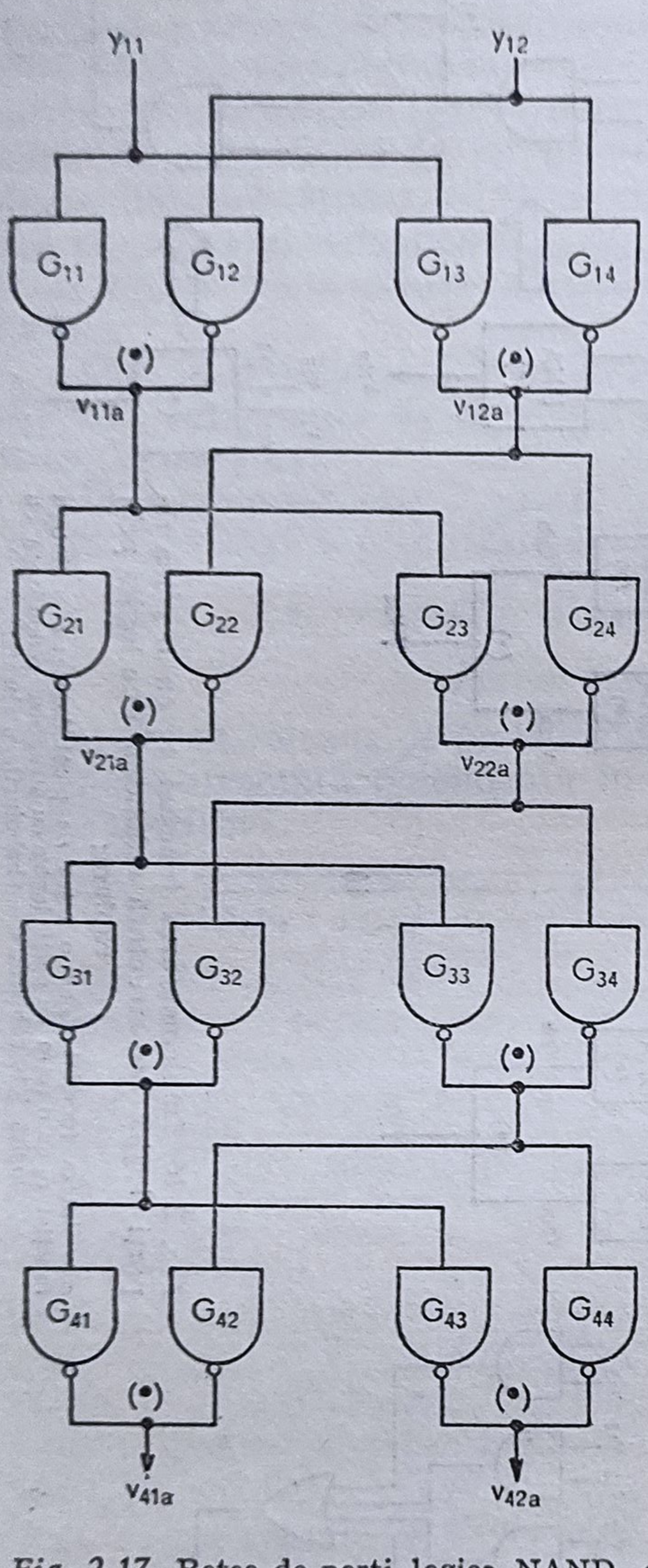


Fig. 2.17. Rețea de porți logice NAND cu structură redondantă cu logică prin cablare.

Mascarea defectărilor multiple cu o tehnică redondantă cu logică prin cablare are cîteva avantaje față de redondanța cu logică cuadruplă sau logică majoritară. Mai întîi, numărul de interconectări între niveluri este mai mic la o rețea redondantă cu logică prin cablare decît la celelalte două tipuri de structuri aplicabile circuitelor logice. În al doilea rînd, schimbarea ordinului redondanței se poate realiza cu ușurință, adăugîndu-se o poartă pe grupare DNA (DNO) sau o grupare identică pentru a crește gradul de protecție a sistemului.

De menționat însă că, în cazul mascării defectărilor multiple din sistem, apare o limitare practică dată de factorul „fan-out" al porților utilizate.

În sfîrșit, un avantaj important al structurilor cu logică prin cablare în mascarea erorilor multiple constă în utilizarea numai de porți simple, de tip NAND și NOR. Pentru structurile redondante cu logică majoritară de tip NMR apare necesitatea utilizării unor votere complicate, care ridică ele însele probleme de fiabilitate.

În concluzie, unul dintre efectele conectării în comun a ieșirilor porților logice constă în reducerea numărului acestora. Acest tip de conexiune constituie baza pentru o structură redondantă interesantă ale cărei avantaje, menționate anterior, evidențiază că logica prin cablare poate fi utilizată în realizarea de sisteme tolerante la defectări, construite cu porți logice.

2.2.5. STRUCTURA REDONDANTĂ PRIN CODARE

Aplicîndu-se pentru codificarea stărilor unui automat un cod redondant — cu biți de informație și control — automatul va avea o structură redondantă, numită în cele ce urmează *redondanță prin codare*.

Implementarea redondanței prin codare unui sistem nu necesită, spre deosebire de celelalte tipuri de redondanță, multiplicarea completă a componentelor acestuia, dar implică utilizarea unor elemente de circuit adiționo-

male, care permit reconstituirea semnalului de ieșire corect în prezența unor defecte ale elementelor sistemului.

Desigur, implementarea redondanței prin multiplicarea unităților funcționale de bază — fie că este vorba de o structură redondantă generală sau logică majoritară — este mai simplă și realizează un cîștig apreciabil al fiabilității. Există însă o categorie de coduri decodabile printr-o tehnică de logică majoritară — numite coduri ortogonale [32] — care soluționează problema implementării redondanței prin codare sistemelor, astfel încît aceasta să aducă un cîștig al fiabilității — raportat la creșterea complexității sistemului — mai important decît aplicarea unei structuri redondante realizată printr-o tehnică de multiplicare.

Se consideră un automat finit de tip Mealy (fig. 2.18); M constă dintr-un număr de elemente de memorie, iar S și E sînt circuite combinaționale, care realizează setul de funcții booleene y și z. La timpul t elementul S al sistemului generează starea următoare $\mathbf{Z}(t+1) = \mathbf{Z}'(t)$, ca o funcție de vectorul de intrare, $\mathbf{X}(t)$, și vectorul-stare prezentă al automatului, $\mathbf{Z}(t)$. Funcția de ieșire, $\mathbf{Y}(t)$, este generată de circuitul combinațional E ca o funcție de vectorul semnal de intrare, $\mathbf{X}(t)$, și de vectorul de stare, $\mathbf{Z}(t)$.

Ecuațiile ieșirii automatului sînt de forma:

$$\begin{aligned} y_1 &= y_1(x_1, x_2, \ldots, x_r, z_1, z_2, \ldots, z_k), \\ y_2 &= y_2(x_1, x_2, \ldots, x_r, z_1, z_2, \ldots, z_k), \\ &\vdots \\ y_m &= y_m(x_1, x_2, \ldots, x_r, z_1, z_2, \ldots, z_k), \end{aligned} \tag{2.8}$$

iar ecuațiile stării următoare — de forma:

$$\begin{aligned} z_1' &= z_1'(x_1, x_2, \ldots, x_r, z_1, z_2, \ldots, z_k), \\ z_2' &= z_2'(x_1, x_2, \ldots, x_r, z_1, z_2, \ldots, z_k), \\ &\vdots \\ z_k' &= z_k'(x_1, x_2, \ldots, x_r, z_1, z_2, \ldots, z_k), \end{aligned} \tag{2.9}$$

unde $x_1, x_2, \ldots, x_r$ sînt semnalele de intrare, $z_1, z_2, \ldots, z_k$ — variabilele de stare prezentă, iar $y_1, y_2, \ldots, y_m$ — semnalele de ieșire ale automatului.

Modelul anterior evidențiază cele 2^k stări interne posibile ale automatului. Introducerea unor stări redondante în vederea mascării unor defectări hardware din structura automatului se va face utilizîndu-se — pentru codarea stărilor interne — un cod redondant corector de erori, ceea ce va conduce la creșterea numărului variabilelor de stare z și, evident, a numărului celulelor de memorie corespunzătoare din structura automa-

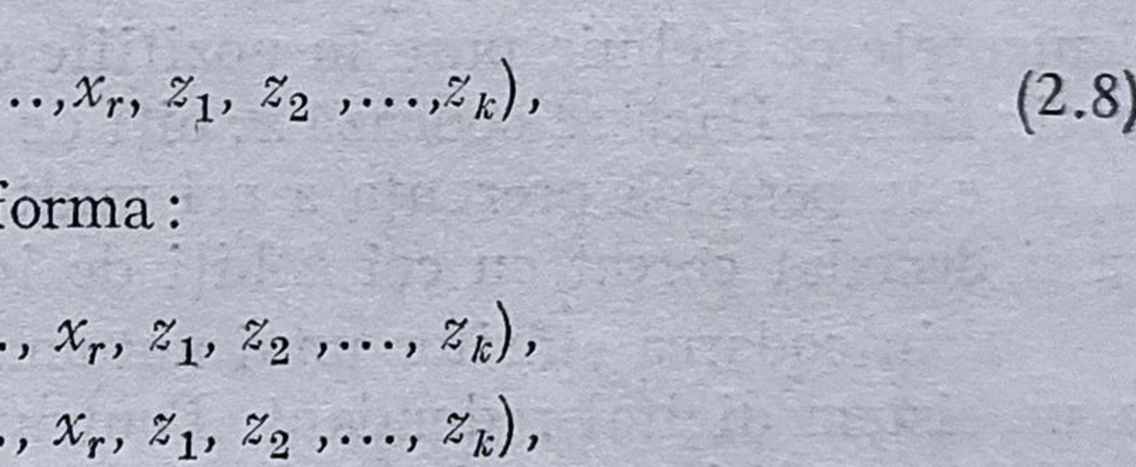


Fig. 2.18. Structura unui automat finit după modelul Mealy.

tului. Dacă numărul variabilelor de stare crește de la k la n, atunci numărul de stări interne va crește de la 2^k la 2^n, astfel încît fiecărei stări a sistemului neredondant îi sînt alocate 2^{n-k} stări. Atunci cînd se va defecta un element hardware din subsistemele care realizează una dintre funcțiile de stare $z_i(t)$, sistemul astfel construit va masca defectul dacă, dintre cele 2^{n-k} stări alocate, sînt stări neafectate de acest defect.

Se consideră o variabilă discretă, x, definită prin trei variabile binare, x_1, x_2, x_3; utilizîndu-se o decizie majoritară simplă, este admisă o eroare la una dintre cele trei variabile pentru ca starea variabilei x să fie corectă. Astfel, orice variabilă discretă poate fi reprezentată cu un cod de forma (n, k), unde n reprezintă numărul total al biților, iar k — numărul biților de informație. În exemplul considerat $n = 3$ și $k = 1$. Codul ales corectează o eroare.

În general, un cod de tipul (n, k) corectează l erori, $l < \frac{n-k}{2}$, $\frac{n-k}{2}$, n și l fiind întregi. Diferența $n - k = m$ reprezintă numărul de poziții disponibile în cuvîntul de cod, numite poziții de control. Simbolurile introduse pe cele m poziții, denumite poziții de control, depind de simbolurile care se găsesc pe pozițiile de informație. Se definește în acest fel un cod sistematic.

Dacă legea de formare a codului se reduce la combinații liniare care permit să se determine — pornind de la elementele pozițiilor de informație— elementele ce trebuie puse în pozițiile de control, se va obține un cod sistematic liniar. Dacă se codează k, biți de informație cu un astfel de cod de tip (n, k) este permisă prezența a cel mult l erori în cuvîntul recepționat, care va fi decodat corect cu cei k biți de informație.

Prin codarea stărilor unui automat finit cu un astfel de cod, se ajunge la un sistem digital redondant. Pentru ca ieșirile sistemului să fie aceleași ca și în cazul sistemului neredondant (număr, semnale), circuitele E_t și S_t, care realizează funcțiile de tranziție, vor fi modificate corespunzător față de cele ale sistemului inițial.

Pentru prezentarea unor particularități ale implementării acestei structuri redondante, se adoptă modelul automatelor finite din figura 2.19*a* [48].

Modelul, o formă detaliată a modelului Mealy, permite o descriere simplă și completă a oricărui sistem digital [40]. În figura 2.19*a*, prin semnalul T s-a reprezentat explicit semnalul de tact care implementează timpul discret al automatului, M_1, M_2,...,M_k sînt celulele de memorie binare, iar C_1, C_2,...,C_k — circuite combinaționale, care realizează funcțiile de tranziție ale automatului. Pentru simplificare, circuitele E din figura 2.18 au fost incluse în circuitele C.

Acest model, atribuit unui sistem digital, prezintă avantajul că evidențiază clar modul în care se implementează tehnica de redondanță prin codare unui sistem hardware. Dacă numărul celulelor de memorie M crește de la k la n, atunci numărul de stări interne va crește de la 2^k la 2^n, fiecare celulă de memorie avînd două stări. În funcționarea automatului vor apărea

stări redondante, care vor determina o funcționare corectă în prezența defectelor hardware, atunci cînd pentru codificarea stărilor se folosește un cod corector de erori.

În figura 2.19*b* este indicată schema unui sistem digital cu structură redondantă, obținută pe baza acestui model. Pentru ca ieșirile să fie aceleași ca în cazul sistemului neredondant (număr, semnale), circuitele C_i sînt modificate față de cele ale sistemului de bază, motiv pentru care ele apar pe schemă notate cu C_i^*.

În modelul de implementare stabilit, stările automatului se codează cu un cod liniar de tip (n, k) cu primii k biți — biți de informație, iar cei-lalți n-k biți — biți redondanți, astfel încît vor fi corectate l erori.

Cu aceste considerente, circuitele C_i^*, $1 \leqslant i \leqslant k$, care generează biții de informație, vor consta dintr-un decodor și un circuit C_i (fig. 2.20*a*). Circuitele C_i^*, $k+1 \leqslant i \leqslant n$, generează biții de control din semnalul stării următoare. Cei n biți ai semnalului $Z(t)$ vor fi prezenți la intrarea circuitului C_i^* și decodați, obținîndu-se o submulțime de k biți informaționali. Acești biți se găsesc la intrările circuitelor $C_{j1}^{(i)}, \ldots, C_{jk}^{(i)}$, care împreună cu codorul formează circuitul C_i^* (fig. 2.20*b*). În acest fel, circuitele C_i^* vor decoda corect cei n biți de intrare și vor genera un semnal de ieșire corect numai atunci cînd apar mai puțin de l erori, numărul de erori apriori fiind fixat în funcție de codul folosit.

Pentru cazurile concrete, aceste circuite pot fi proiectate după una dintre metodologiile specifice sistemelor secvențiale sincrone sau asincrone. Structura circuitelor *decodor*, *codor*, a circuitelor C_i, C_{jk} va fi funcție de tipul codului utilizat.

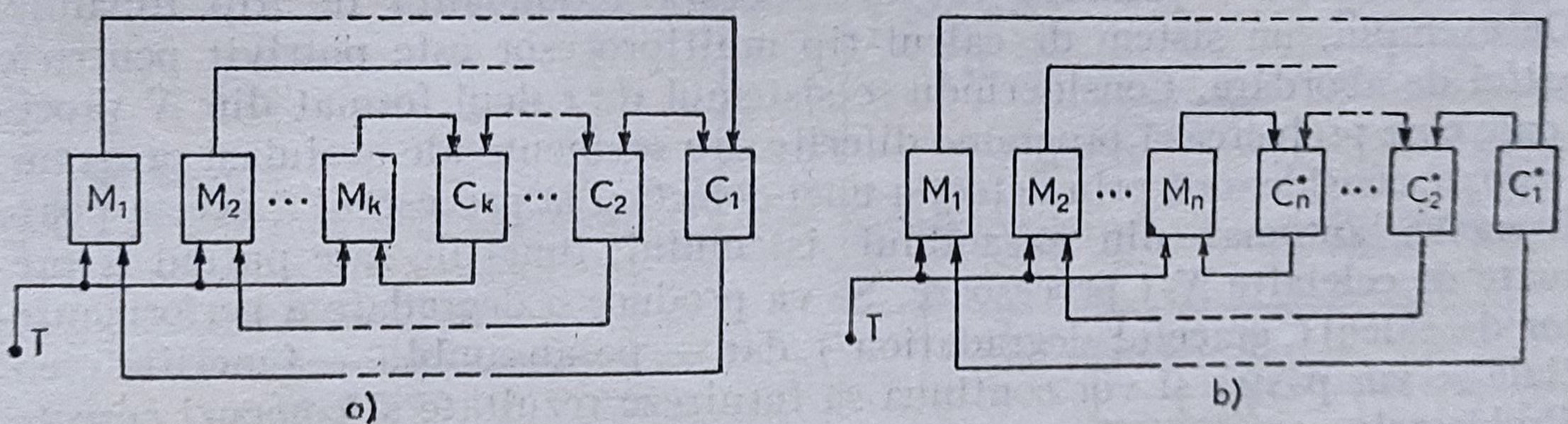


Fig. 2.19. Implementarea unei codări redondante la un automat finit:

a — automatul finit neredondant; *b* — automat finit cu structură redondantă.

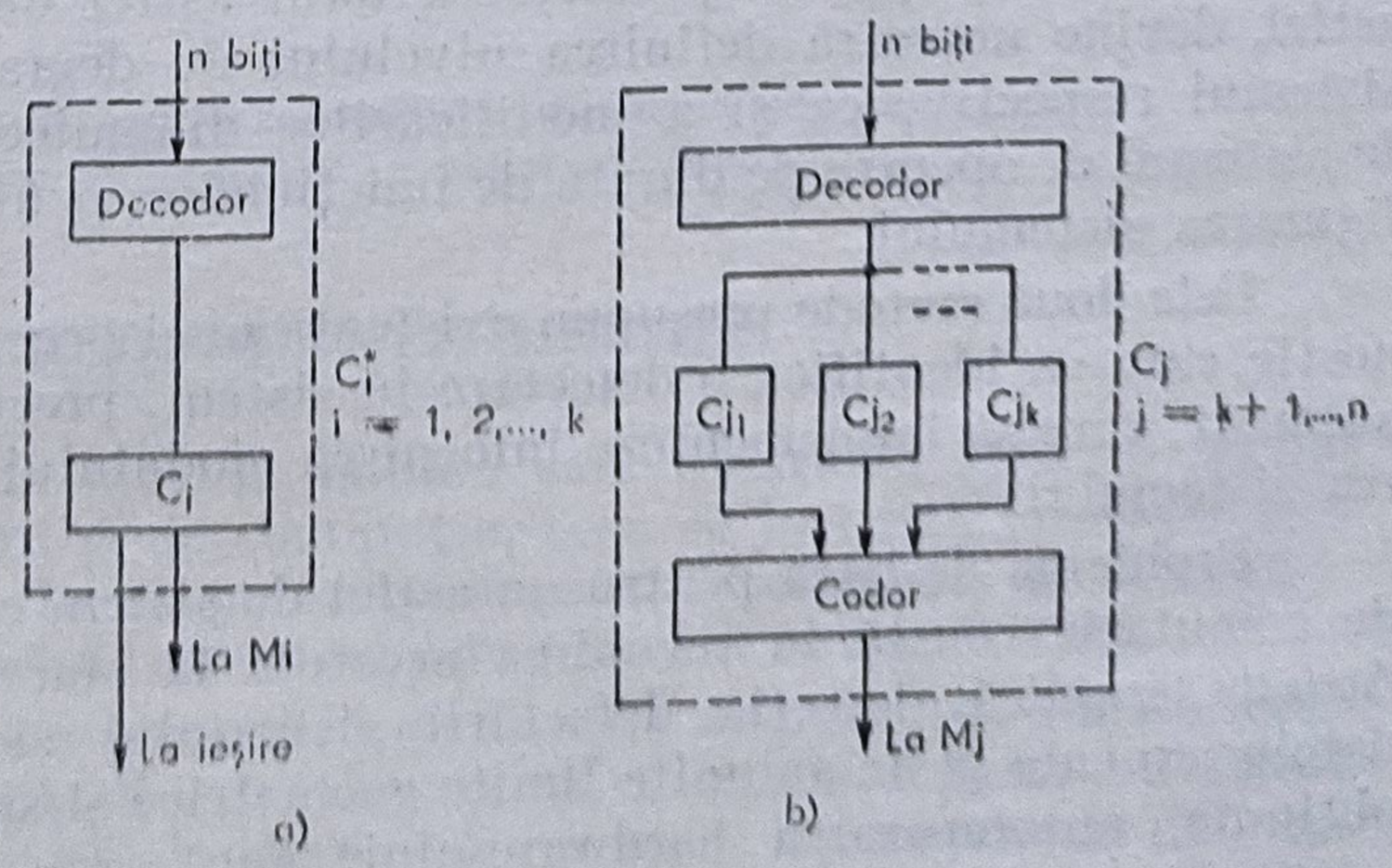


Fig. 2.20. Structura circuitelor C_i^*:

a — cazul $1 \leqslant i \leqslant k$; *b* — cazul $k+1 \leqslant i \leqslant n$.

Se remarcă faptul că pentru acest tip de redondanță, spre deosebire de tipurile de redondanță rezultate prin multiplicarea componentelor, intervin modificări în structura sistemului de bază.

2.2.6. STRUCTURI REDONDANTE PROTECTIVE DINAMICE

Acest tip de structură redondantă („standby redundancy") utilizează cîteva module identice din punct de vedere funcțional, unul sau o parte dintre module fiind active, realizînd funcția sistemului, celelalte fiind în așteptare, urmînd a fi comutate, atunci cînd unul dintre modulele active va „cădea". De aceea structura se numește și structură redondantă de *comutație*.

În acest caz tolerarea defectului este explicită: la ieșirea sistemului/modulului defect apare secvența de semnale eronate, care este detectată și corectată prin înlocuirea sistemului/modulului defect — în mod automat — cu modulele aflate în așteptare. Metoda este similară acțiunii de SERVICE în vederea reparării. Prin acest mod de acționare — de înlocuire automată a modulului defect cu un modul identic în stare de bună funcționare — structura redondantă protectivă dinamică realizează un *sistem autoreparabil*. Aceasta face ca respectiva structură să fie utilizată în sistemele cu misiuni de lucru de lungă durată și la care intervenția omului în vederea reparării nu este posibilă sau nu este permisă.

De menționat că sistemele reconfigurabile — sistemele care își modifică structura la detectarea unei defectări prin înlocuirea modulelor defecte cu alte module mai simple, astfel încît degradarea performanțelor să fie de un nivel admisibil — prezintă tot o structură redondantă de tip dinamic. De exemplu, un sistem de calcul tip multiprocesor este potrivit pentru o astfel de abordare. Considerîndu-se sistemul de calcul format din N procesoare care prelucrează programe diferite sau segmente ale aceluiași program, se poate considera că prin detecția unei defectări la procesorul i, acesta poate fi exclus automat din ansamblul sistemului, funcțiile sale putînd fi preluate de celelalte N-1 procesoare. Se va produce o degradare a performanțelor de calcul („graceful degradation"), dar — pe ansamblu — funcțiile esențiale se vor păstra și vor continua să furnizeze rezultate și comenzi corecte. Problemele care apar la proiectarea unui astfel de sistem sînt complexe; astfel, devine necesară definirea nivelului de degradare admisibilă pentru sistemul respectiv, ca și a modificărilor dinamice care trebuie efectuate în sistemul de operare pe durata de funcționare, pentru a supraveghea reconfigurarea sistemului.

Cele două metode presupun existența unei proceduri automate de diagnostic, care va identifica o defectare în sistem, precum și a unui comutator automat, care să implementeze înlocuirea modulului defect sau reconfigurarea sistemului.

Problema de bază pentru un astfel de sistem cu structură redondantă de comutație constă în revenirea acestuia la starea de bună funcționare. Aceasta implică detecția defectării sistemului și prevenirea propagării datelor eronate peste anumite limite geometrice și/sau controlul informației obținute, autorepararea hardware-ului, dacă este necesar, reconstituirea

informației afectate de eroare, reluarea funcțiilor sistemului. Această structură se aplică facil sistemelor organizate modular. Modulele trebuie astfel alese încît să poată fi diagnosticat cu ușurință modulul defect.

În figura 2.21 *a* se prezintă un sistem de calcul cu structură redondantă de comutație în configurație de sistem autoreparabil. Unul din cele două sisteme funcționează *on-line*, iar celălalt — *off-line*, fiind rezerva primului sistem și conectat doar în momentul defectării acestuia.

În figura 2.21 *b* se prezintă o variantă de sistem reconfigurabil. Sistemul este format din patru unități A_i și B_i ($i = 1,2$); uzual toate cele patru unități funcționează *on-line*. Atunci cînd este detectată defectarea uneia dintre unitățile A_i sau B_i, sistemul își continuă funcționarea fără unitatea defectă, reconfigurîndu-se astfel într-un sistem cu performanțe de fiabilitate mai slabe. Evident, dacă două unități A_i sau B_i ($i = 1,2$) se defectează simultan, atunci sistemul se defectează.

Un astfel de sistem necesită un software și un hardware complexe, dar prezintă performanțe de fiabilitate ridicate.

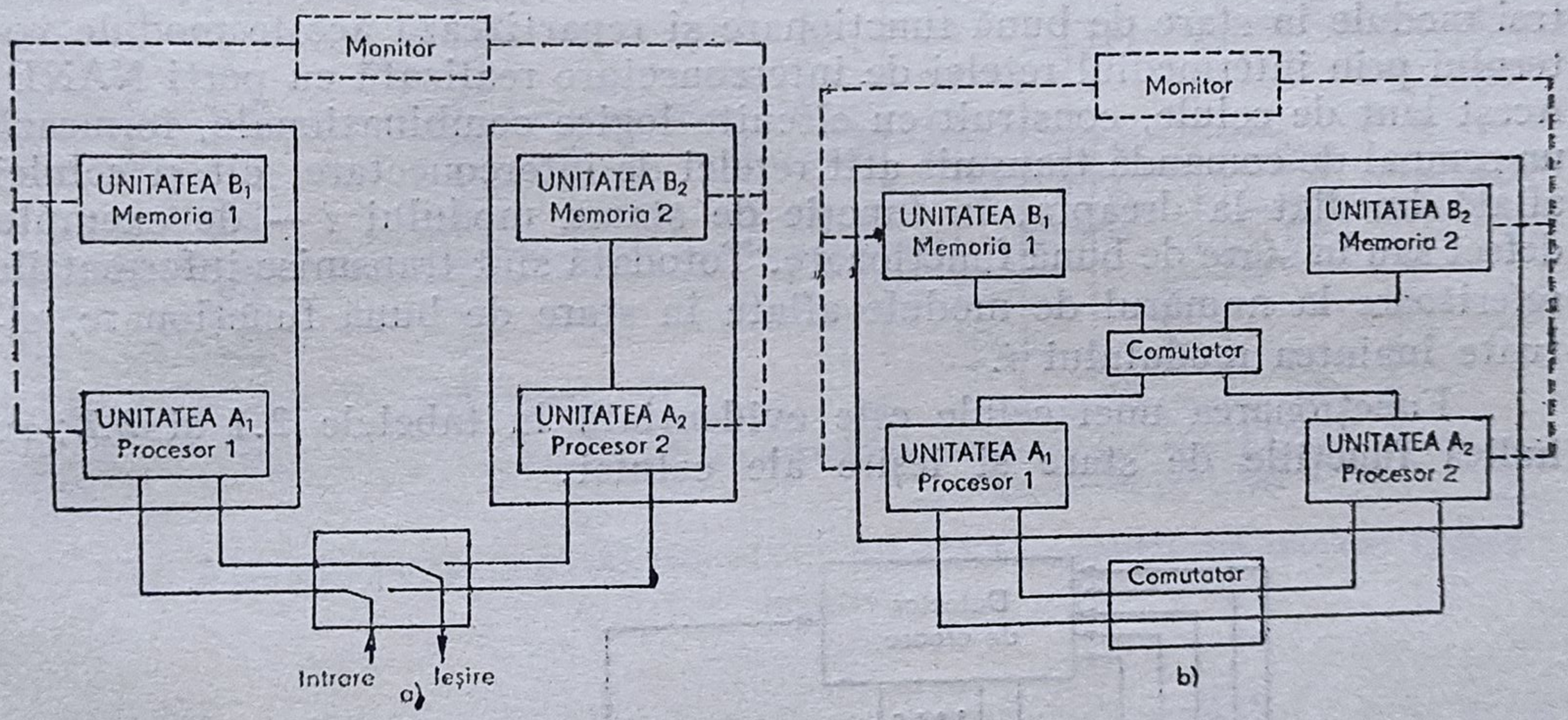


Fig. 2.21. Sisteme cu structură redondantă dinamică (de comutație):
a — sistem autoreparabil; *b* — sistem reconfigurabil.

2.2.7. STRUCTURI REDONDANTE HIBRIDE

Structura redondantă hibridă îmbină caracteristicile structurilor redondante logică majoritară și de comutație, ceea ce justifică interesul special pentru utilizarea sa în scopul implementării toleranței la defectări. Schema tipică a unei astfel de structuri este indicată în figura 2.22.

Structura are la bază un nucleu de $2n + 1$ module funcționale identice, conectate prin intermediul rețelei de interconectare pentru a forma o structură logică majoritară de tip $n + 1$ din $2n + 1$ și un număr de r rezerve care urmează să fie conectate în momentul detecției unor defectări la cele $2n - 1$ module. Sistemul detector de eroare testează ieșirile celor

$2n + 1$ module identice ale nucleului evidențiind defectarea unui modul. Rețeaua de interconectare este astfel concepută încît înlocuiește modulul defect cu o rezervă atunci cînd una din cele $2n + 1$ ieșiri testate este eronată.

O astfel de schemă poate avea o fiabilitate foarte înaltă dacă voterul, sistemul de comutație și detectorul de eroare sînt foarte fiabile. Aceasta implică cerința unui sistem de comutație simplu. O soluție în acest sens constă în introducerea comutatoarelor celulare iterative [53]. În figura 2.23 se prezintă o structură redondantă hibridă cu comutator celular iterativ. Cele cinci circuite basculante bistabile, CBB_c, memorează necoincidența între ieșirea unui modul și ieșirea voterului. Detecția neconcordanței este realizată de circuitele SAU EXCLUSIV la intrarea cărora apar semnalele de ieșire de la modulele funcționale și de la voter. Același impuls de tact care acționează asupra modulelor funcționale este utilizat și pentru acționarea circuitelor basculante bistabile, CBB_c. Întîrzierea introdusă este necesară pentru a realiza sincronizarea funcționării sistemului, egalînd întîrzierea fizică a semnalelor prin porțile rețelei de interconectare și voter.

Lanțul de celule iterative este astfel proiectat încît determină primele trei module în stare de bună funcționare și repartizează aceste module voterului prin intermediul rețelei de interconectare realizată cu porți NAND. Acest lanț de celule, construit cu circuite logice combinaționale, formează un semnal de comandă transmis atît rețelei de interconectare, cît și celulei aflată imediat la dreapta, în funcție de starea modului i — de exemplu defect sau în stare de bună funcționare. Totodată sînt transmise informațiile referitoare la numărul de module aflate în stare de bună funcționare, situate înaintea modulului i.

Funcționarea unei celule este evidențiată în tabelele 2.1 și 2.2; se indică funcțiile de stare și ieșire ale celulei.

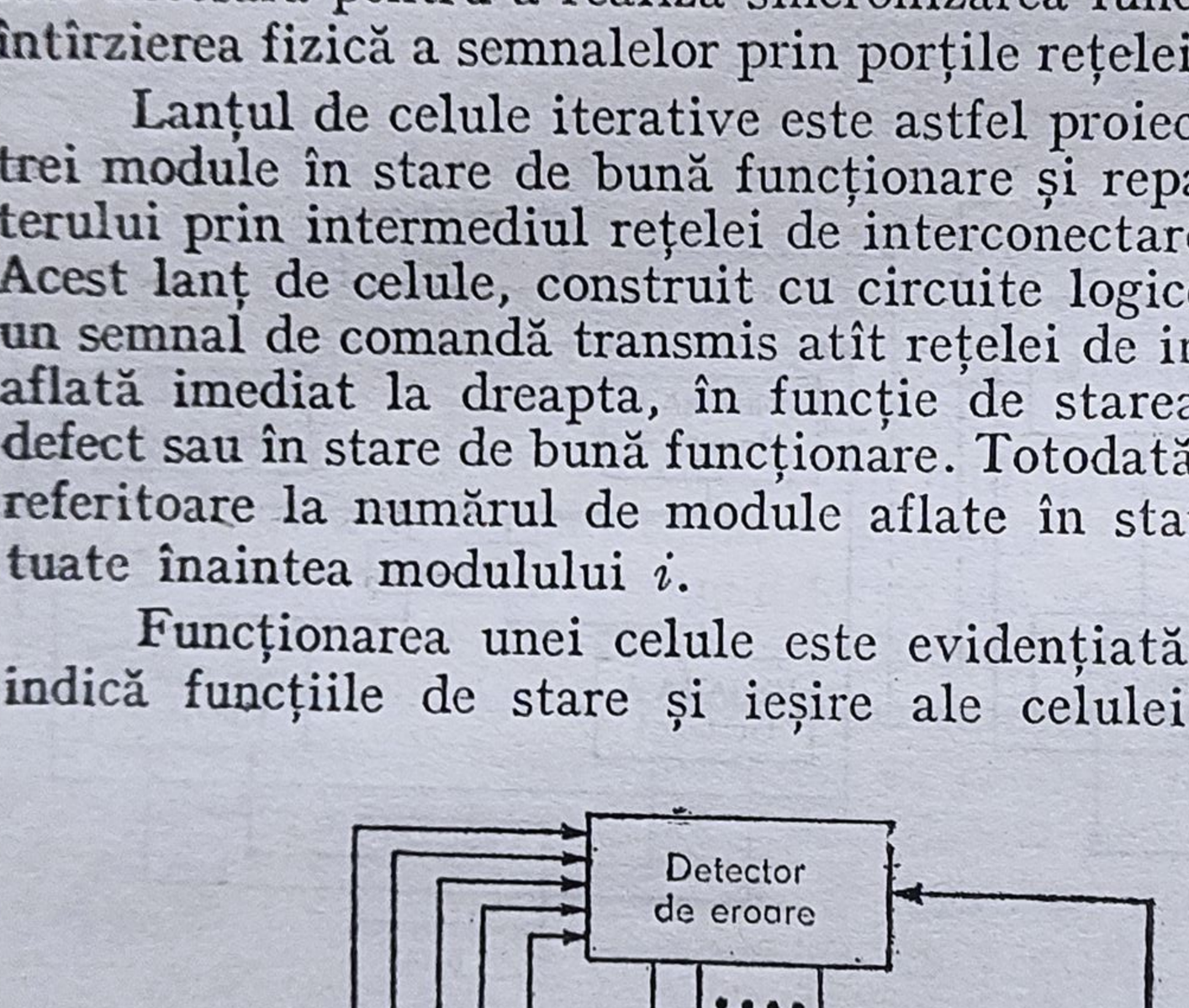


Fig. 2.22. Structura redondantă hibridă.

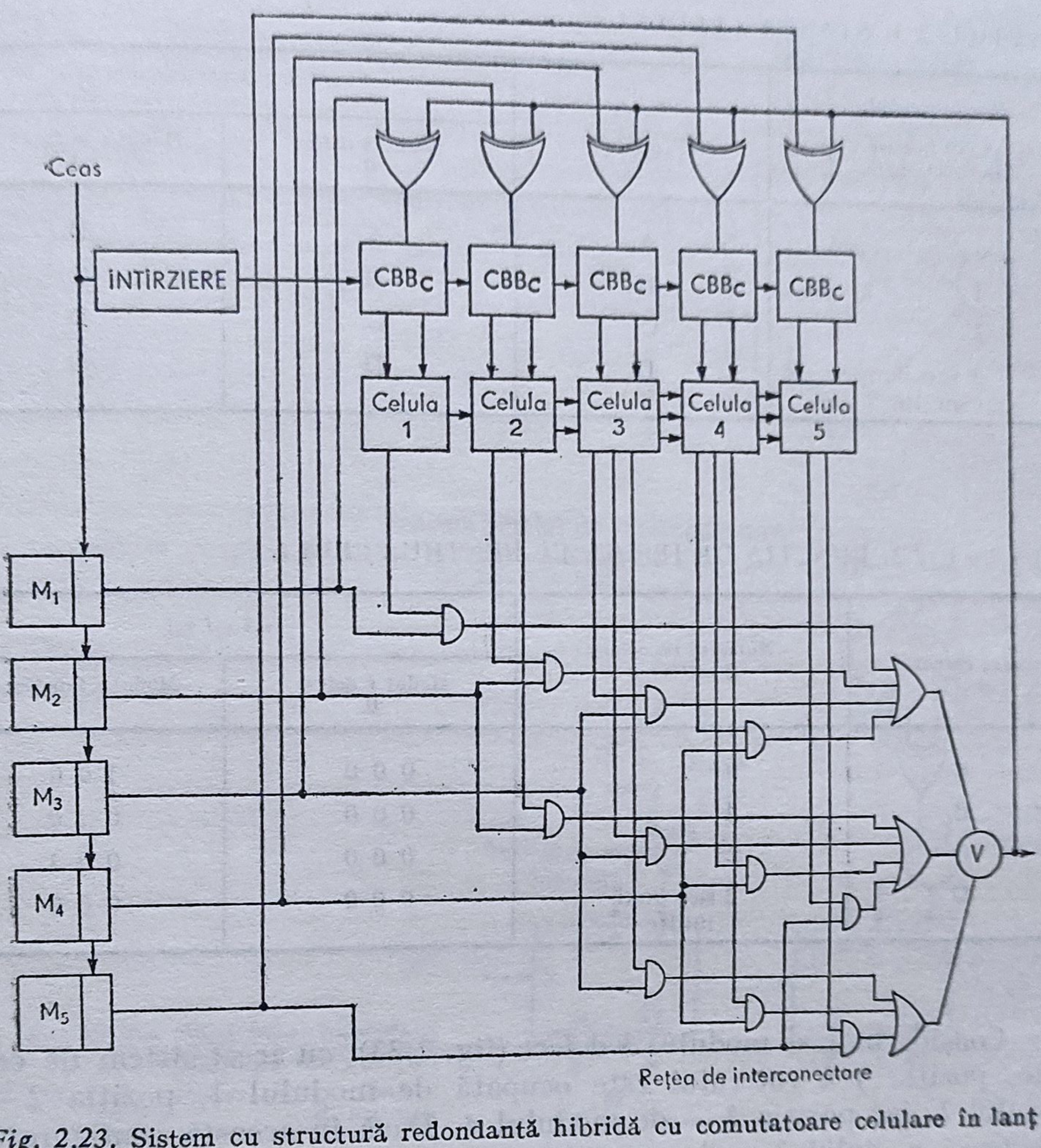


Fig. 2.23. Sistem cu structură redondantă hibridă cu comutatoare celulare în lanț iterativ.

Din inspecția celor două funcții rezultă că dacă modulul i este defect, informația referitoare la numărul modulelor aflate în stare de bună funcționare trece prin celulă fără a suferi vreo modificare. În caz contrar, dacă modulul i este în stare de bună funcționare, are loc incrementarea cu 1 a stării celulei respective. Dacă la un moment dat există numărul complet de module repartizat voterului V, nu mai este necesară nici o iterație.

Semnalele de ieșire V^i ale celulei C_i repartizează modulul i la intrarea voterului V. De exemplu, dacă semnalul de la stînga celului indică că ar fi un modul în stare de funcționare, iar celula C_i indică funcționarea modulului i, atunci starea 2 este transmisă la dreapta. Ieșirea va indica 1 pe linia V_2^i, care comandă rețelei de interconectare să repartizeze modulul i la poziția a doua a voterului, pe celelalte linii apărînd semnal 0. Poziția 1 a voterului este deja ocupată de un alt modul aflat în stare de funcționare (situat înaintea modulului i).

Tabelul 2.1. STAREA CELULEI C_i

Număr module anterioare aflate în stare de funcționare	Starea curentă a celulei i	Starea următoare a celulei C_i	
		Modul i defect 0	Modul i în funcțiune 1
0	A	A	B
1	B	B	C
2	C	C	D
3 sau mai multe	D	D	D

Tabelul 2.2. FUNCȚIA DE IEȘIRE V^i PENTRU CELULA C_i

Starea curentă	Numărul de celule funcționale	$v_1 i\ v_2 i\ v_3 i$	
		Modul i defect 0	Modul i funcțional 1
A	0	0 0 0	1 0 0
B	1	0 0 0	0 1 0
C	2	0 0 0	0 0 1
D	3 sau mai multe	0 0 0	0 0 0

Considerîndu-se modulul 3 defect (fig. 2.23), cu acest sistem de comutație, poziția 1 a voterului este ocupată de modulul 1, poziția 2 — de modulul 2, iar poziția 3 — de modulul 4. Dacă în această configurație se defectează modulul 2 — de exemplu —, atunci pozițile 1, 2 și 3 ale voterului vor fi ocupate respectiv de modulele 1, 4 și 5. Rețeaua de interconectare considerată în fig. 2.23 realizează o conectare logică a modulelor M_i prin intermediul rețelei de porți *ȘI* (fig. 2.24).

Atunci cînd apare o necoincidență între ieșirea unui modul și ieșirea voterului, circuitul basculant bistabil de poziție, CBB_c — care poate fi realizat cu circuite basculante bistabile de tip JK — este adus în starea „0" logic, altfel el rămîne în starea „1" logic (fig. 2.25 *a*). Necoincidențele sînt astfel recepționate de îndată ce au loc, iar modulele defecte sînt înlocuite într-un ciclu de funcționare al generatorului de tact al sistemului.

Există varianta ca modulele de rezervă, în așteptare, să nu fie alimentate. În ipoteza că modulele neîncărcate electric au pe liniile de ieșire semnalul logic „0", celula C_i va conecta modulul M_i în sistemul *NMR* prin intermediul unui comutator de putere din modul, utilizîndu-se aceeași rețea de interconectare (fig. 2.25 *b*). La intrarea circuitului basculant de poziție se introduce o poartă ȘI suplimentară față de schema din fi-

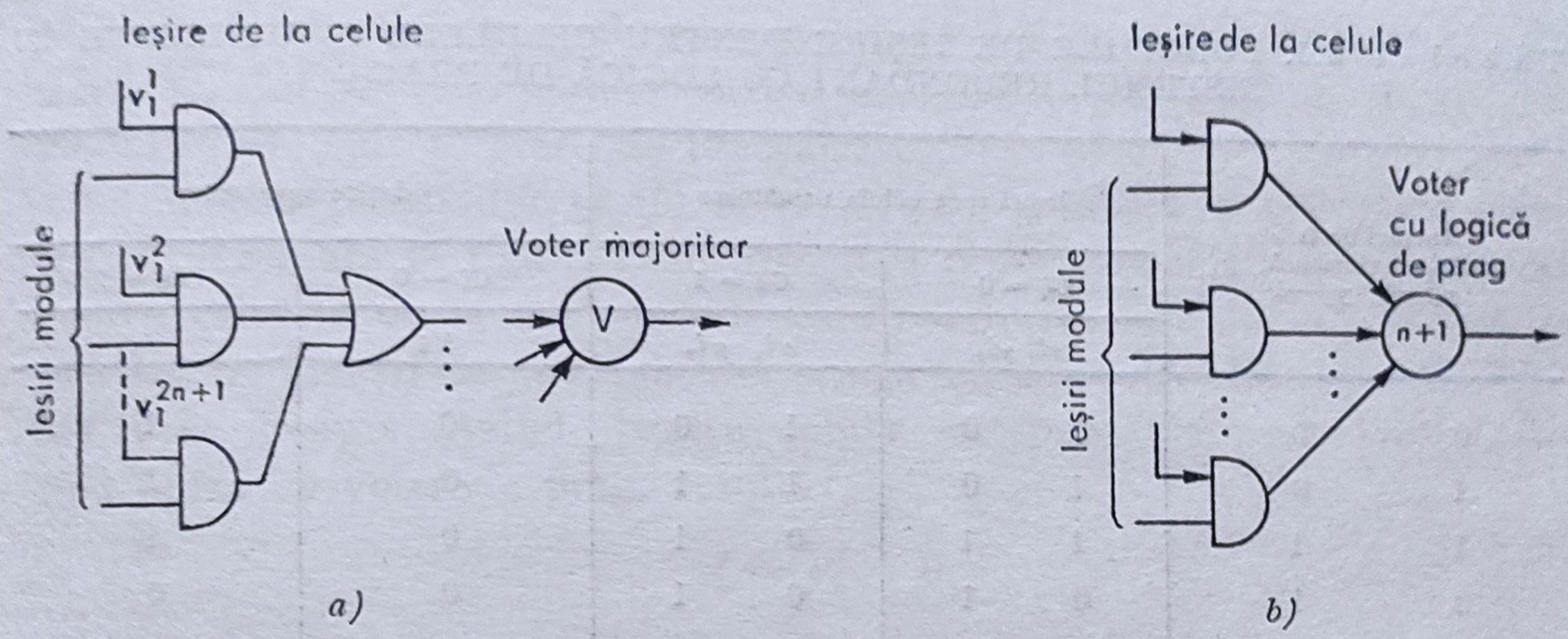


Fig. 2.24. Schema reţelei de interconectare:
a — în cazul utilizării sistemului de decizie cu logică majoritară: *b* — în cazul utilizării sistemului de decizie cu logică de prag.

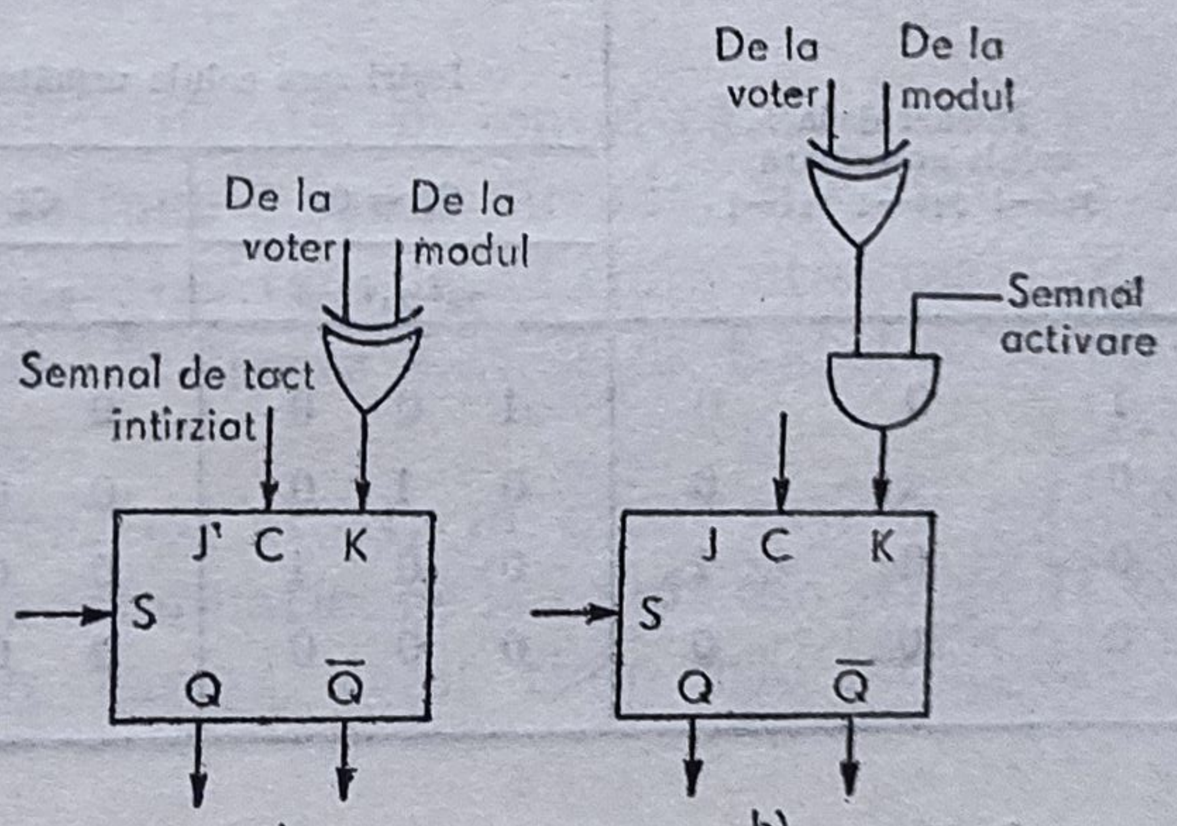


Fig. 2.25. Schema circuitului basculant bistabil de poziţie:
a — fără semnal de activare; *b* — cu semnal de activare.

gura 2.25 *a*, pentru ca rezervele neîncărcate energetic să nu fie interpretate eronat ca fiind defecte (fig. 2.25 *b*). De menţionat că o rezervă neîncărcată se poate defecta fiind în aşteptare, dar bistabilul de condiţie, CBB_e, nu va recepţiona această defectare pînă ce modulul corespunzător nu este comutat în sistemul *NMR*. La impulsul următor de tact, rezerva defectă va fi eliminată. Deci, pentru o schemă cu rezervare pasivă, sînt necesare două perioade de tact consecutive, care să detecteze şi să deconecteze rezerva defectă.

Voterul din figura 2.23 poate fi realizat fie după o logică majoritară, fie după o logică de prag [26] cu valoarea de prag n şi cu $2n + 1 + r$ intrări. În figura 2.24 *b* este indicată o schemă de interconectare logică pentru voterul cu logică de prag. Celula C_i este mai simplă decît în cazul unui voter majoritar, deoarece va forma un semnal care va condiţiona ca ieşirea modulului să fie votată sau nu. Această simplificare a lanţului de celule şi a reţelei de interconectare este compensată de un sistem de votare mai complex decît în cazul logicii majoritare.

Tabelul 2.3. FUNCȚIILE DE IEȘIRE PENTRU CELULELE C_i UTILIZATE ÎN SISTEMUL REDONDANT CU LOGICĂ DE PRAG

Intrări de la celula anterioară y_1^{i-1} y_2^{i-1}	Ieșiri spre celula următoare				Ieșire spre voter	
	$Ci = 0$		$Ci = 1$		$Ci = 0$	$Ci = 1$
	yi_1 yi_2		yi_1 yi_2		Vi	Vi
0 0	0	0	1	0	0	1
1 0	1	0	1	1	0	1
1 1	1	1	0	1	0	1
0 1	0	1	0	1	0	0

Tabelul 2.4. FUNCȚIILE DE IEȘIRE PENTRU CELULELE C_i UTILIZATE ÎN SISTEMUL REDONDANT CU LOGICĂ MAJORITARĂ

Intrări de la celula anterioară y_0^{i-1} y_1^{i-1} y_2^{i-1}	Ieșiri spre celula următoare						Ieșiri spre voter					
	$C_i = 0$			$C_i = 1$			$C_i = 0$			$C_i = 1$		
	y_0^i	y_1^i	y_2^i	y_0^i	y_1^i	y_2^i	v_1^i	v_2^i	v_3^i	v_1^i	v_2^i	v_3^i
1 0 0	1	0	0	0	1	0	0	0	0	1	0	0
0 1 0	0	1	0	0	0	1	0	0	0	0	1	0
0 0 1	0	0	1	0	0	0	0	0	0	0	0	1
0 0 0	0	0	0	0	0	0	0	0	0	0	0	0

În continuare, se va exemplifica realizarea unei celule C_i în cele două cazuri: voter cu logică de prag și, respectiv, cu logică majoritară.

Pe baza considerațiilor de mai sus, în tabelele 2.3 și 2.4 se dau semnalele funcțiilor de ieșire ale celulelor pentru cele două situații — sistem de decizie cu logică de prag și, respectiv, cu logică majoritară, rezultînd ecuațiile logice corespunzătoare:

$$\begin{aligned} y_1^i &= \overline{C}_i\, y_1^{i-1} + C_i\, \overline{y}_2^{i-1}, \\ y_2^i &= y_2^{i-1} + C_i\, y_1^{i-1}, \\ v^i &= C_i\, y_1^{i-1} + C_i\, \overline{y}_2^{i-1}, \end{aligned} \tag{2.10}$$

pentru celulele din sistemul redondant cu voter cu logică de prag, și

$$\begin{aligned} y_0^i &= \overline{C}_i\, y_0^{i-1}, \\ y_1^i &= \overline{C}_i\, y_1^{i-1} + C_i\, y_0^{i-1}, \\ y_2^i &= \overline{C}_i\, y_2^{i-1} + C_i\, y_1^{i-1}, \\ v_1^i &= C_i\, y_0^{i-1}, \\ v_2^i &= C_i\, y_1^{i-1}, \\ v_3^i &= C_i\, y_2^{i-1}, \end{aligned} \tag{2.11}$$

pentru celulele din sistemul redondant cu voter cu logică majoritară. Intrările C_i sînt „0" cînd se detectează o defectare a modulului M_i și „1" cînd modulele M_i sînt în stare de funcționare.

Pe baza ecuațiilor (2.10) și (2.11) se pot sintetiza cele două celule tipice prezentate în figura 2.26. Structura prezentată poate fi simplificată pentru celulele de la marginea lanțului, datorită faptului că acestea nu primesc informații de la celula din stînga și, respectiv, nu transmit informații la dreapta. Astfel, pentru celula corespunzătoare modulului M_1, în tehnica cu voter de prag, rezultă:

$$y_1^0 = y_2^0 = 0,$$

iar ecuațiile logice care sintetizează structura celulei sînt:

$$\begin{aligned} y_1^1 &= C_1; \\ y_2^1 &= 0; \\ v^1 &= C_1. \end{aligned} \tag{2.12}$$

De asemenea, celula 2 este simplificată, în sensul că ea nu va primi niciodată informația că sînt în funcțiune 2 sau 3 module, ci maximum un modul. Pentru celula 5 nu se sintetizează funcția de stare următoare, dacă nu există celule la dreapta acesteia. În figura 2.27 se dă lanțul celor cinci celule, cu simplificările amintite.

Soluția aleasă introduce — ca, de altfel, în cazul tuturor rețelelor iterative — un timp de propagare a informației de la celula din stînga la cea din extrema dreaptă. În anumite cazuri, acest timp este prohibitiv de

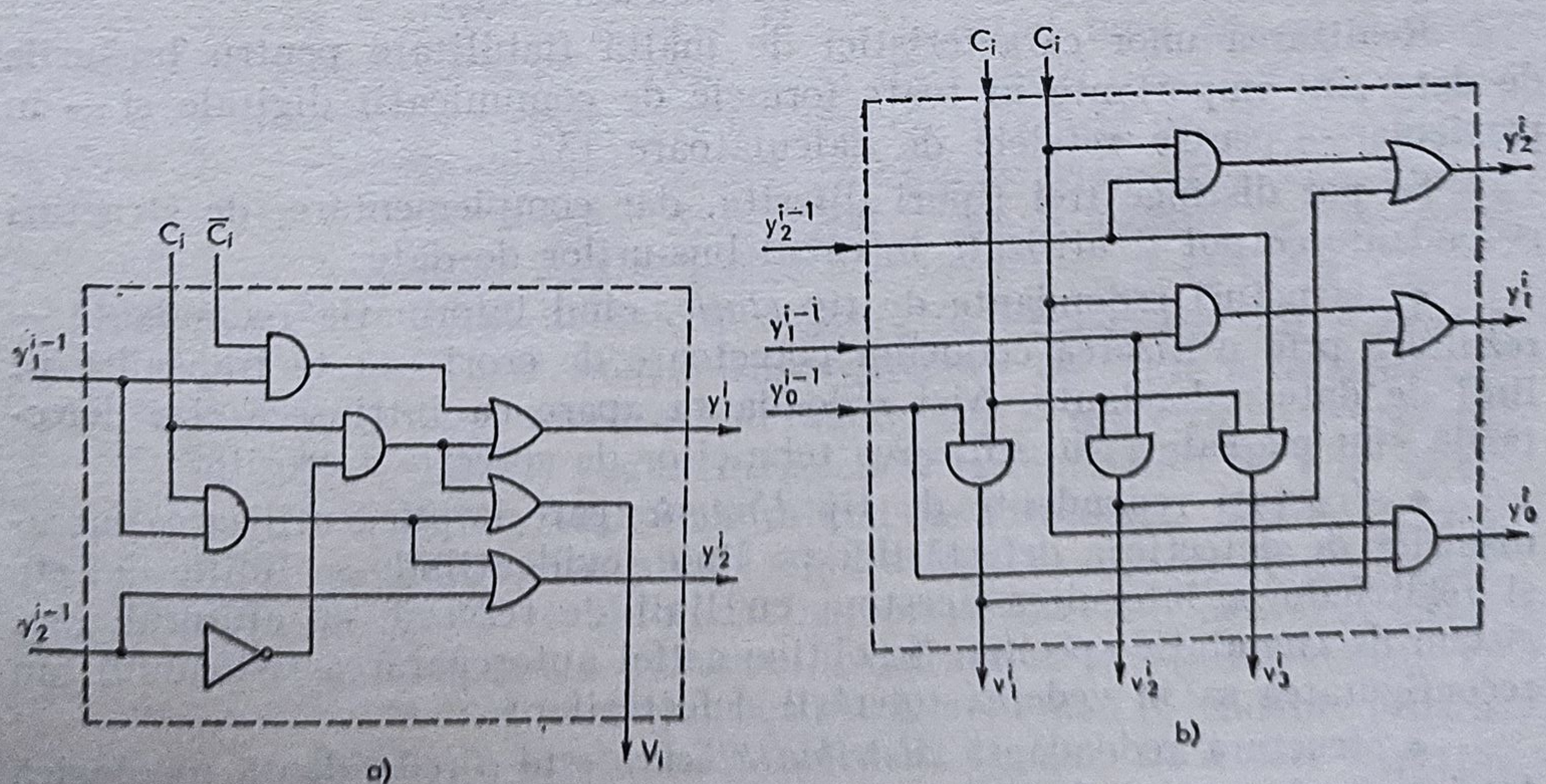


Fig. 2.26. Schemele electrice ale celulelor C :

a — celula C pentru o tehnică de decizie cu logică de prag; *b* — celula C pentru o tehnică de decizie cu logică majoritară.

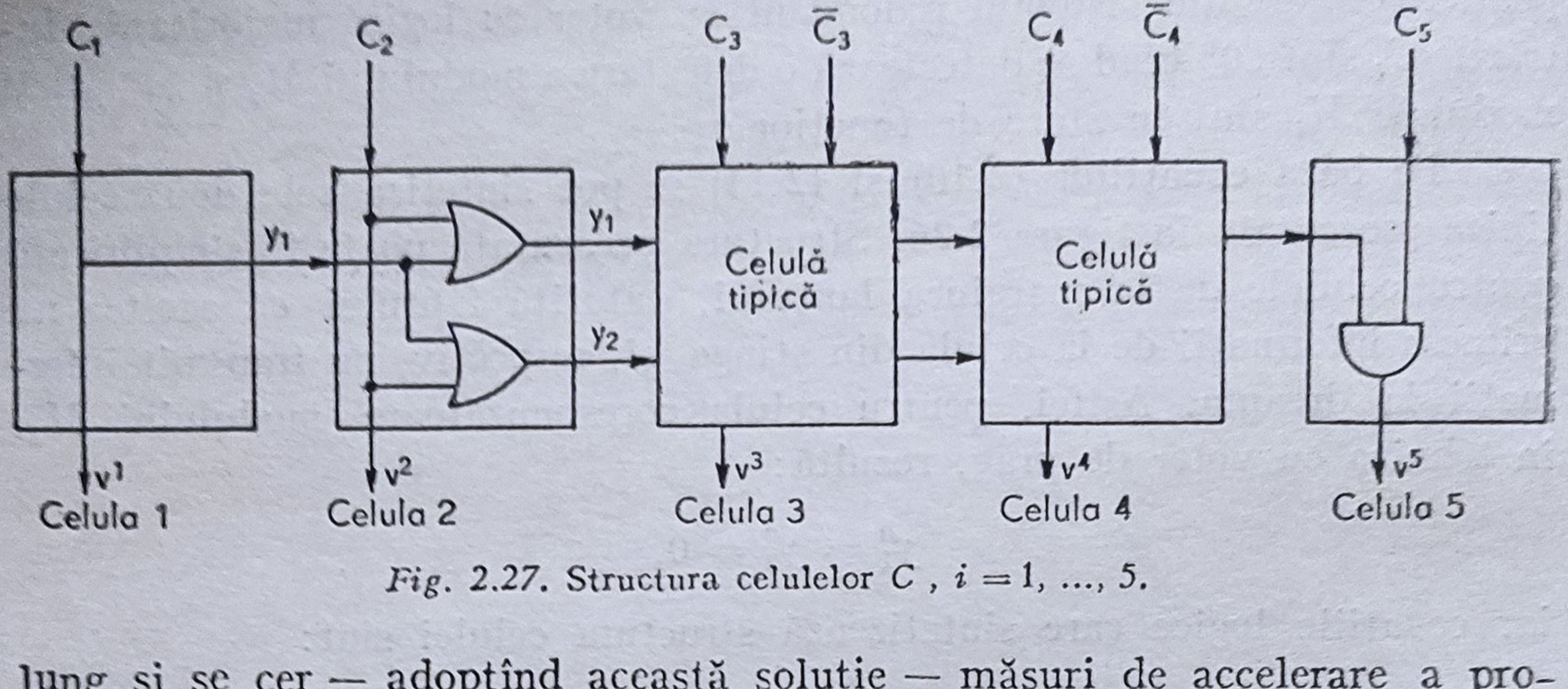


Fig. 2.27. Structura celulelor C_i, $i = 1, \ldots, 5$.

lung și se cer — adoptînd această soluție — măsuri de accelerare a propagării informației. În [53] se prezintă astfel de metode, implementate în hardware-ul sistemului.

2.3. STRUCTURI REDONDANTE PENTRU INTERCONEXIUNILE UNUI SISTEM

În cazul implementării unui sistem de înaltă fiabilitate cu structură tolerantă la defectări, ale cărui elemente sînt protejate la apariția defectărilor prin tehnici de redondanță, devine necesar ca și interconexiunile dintre elementele sistemului să admită o toleranță la defectările posibile, chiar dacă ele sînt considerate mult mai fiabile decît celelalte elemente ale sistemului. Acest lucru este în particular necesar pentru sistemele care utilizează un bus comun de date între modulele sale.

Realizarea unor caracteristici de înaltă fiabilitate pentru bus-urile de date sînt importante în toate formele de comunicații digitale și — în particular — pentru rețelele de calculatoare [37].

Se pot distinge trei tipuri diferite, dar complementare, de structuri redondante ce pot fi utilizate în cazul bus-urilor de date:

- structuri redondante de tip *static*, cînd informația redondantă — rezultată prin utilizarea codurilor corectoare de erori — este transmisă pe linii de date redondante. Aici redondanța apare ca intrinsecă, iar defectările sînt mascate prin utilizarea tehnicilor de corecție a erorilor;
- structuri redondante de tip *dinamic*, care implică utilizarea mecanismelor de detecție a defectărilor pe linie, evidențiindu-se liniile defecte și realizîndu-se înlocuirea acestora cu linii de rezervă, cu ajutorul unui sistem de comutație specific. Se obține astfel autorepararea sistemului sau reconfigurarea sa în vederea tolerării defectărilor;
- structura redondantă *distribuită*, care este o redondanță topologică intrinsecă sistemului de linii de informație, permițînd o tolerare perfectă a defectărilor prin utilizarea unor rute alternate, dar care conduce la o reducere a performanțelor de operare ale sistemului.

Primele două tipuri de structuri sînt comune cu structurile redondante aplicate modulelor funcționale, în timp ce a treia formă este tipică pentru sistemele de interconectare.

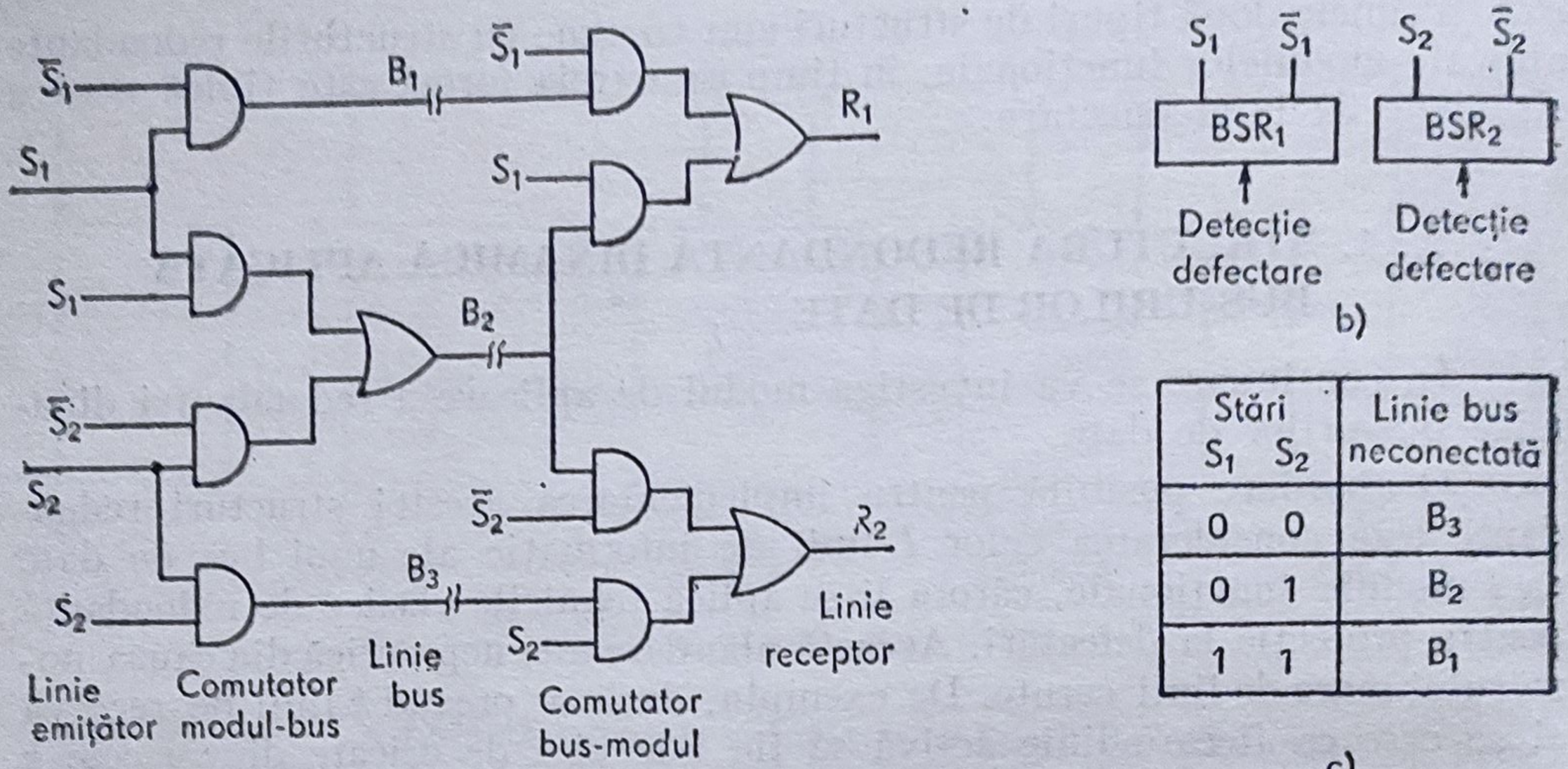
2.3.1. STRUCTURA REDONDANTĂ DINAMICĂ APLICATĂ BUS-URILOR DE DATE

În continuare se va investiga modul de aplicare a redondanței dinamice bus-urilor de date.

O abordare posibilă pentru implementarea acestei structuri redondante este considerarea celor l linii de informație ale unui bus de date ca l module funcționale, cărora li se aplică tehnicile clasice de redondanță pentru protecție la defectări. Această abordare este nepractică din cauza numărului mare de linii cerute. De exemplu, dacă se prevăd k linii de rezervă și se cere ca fiecare linie activă să fie înlocuită de oricare dintre cele k rezerve, atunci comutatoarelor care asigură această reconfigurare li se vor cere caracteristici de fan-in, fan-out și putere excesive [12]. Implicit, într-un astfel de aranjament este necesar un mecanism specific pentru detecția și localizarea defectărilor pe linie.

Dacă însă se cere celor l linii de informație I_j să fie conectate la una dintre liniile de bus, B_j, B_{j+1}, ..., B_{j+k}, va rezulta un sistem de comutație mai simplu. Creșterea în fiabilitate adusă de cele două aranjamente este aceeași. Și într-un caz și în celălalt numărul maxim de stări al unei linii este $k + 1$; dar cerințele de *fan-in* și *fan-out* sînt reduse de la l — în primul caz — și la k în cel de-al doilea caz. Se consideră că l este — de obicei — cu un ordin de mărime mai mare decît k.

Fie următorul exemplu. Informația este transmisă de un modul emițător pe două linii, S_1 și S_2, către două linii ale unui modul receptor, R_1 și R_2. Există o linie de rezervă ($k = 1$), astfel încît fiecare linie de informație va avea două stări. Aceste două stări vor fi memorate de un registru al stărilor bus-ului, BSR, care în cazul de față este un simplu circuit basculant bistabil. Comutatorul va comuta linia de informație S_1 la bus-ul 1 sau 2, iar linia de informație S_2 la busul 2 sau 3, funcție de starea de defect detectată. În figura 2.28 se indică o structură posibilă pentru comutatoarele modul-bus și bus-modul, dezvoltată pe baza modelului abordat. În acest exemplu particular, conținutul registrelor BSR are o interpretare simplă, asemănătoare cu cea indicată în tabelul din figura 2.28 *c*. Se observă că fiecare linie de informație poate fi conectată la două linii de bus. Dacă celula BSR_j este în starea logică „0", linia de informație este conectată la linia de bus superioară, iar atunci cînd celula se află în starea logică „1", linia este conectată la linia de bus următoare. Un defect al sistemului de registre BSR poate conduce la defectarea întregului sistem. O soluție pentru evitarea acestui neajuns este aplicarea unei redondanțe pentru registrele BSR. Astfel, în figura 2.29 cele două registre BSR sînt triplate [42], iar fiecare comutator al schemei precedente este înlocuit de un aranjament de votare din trei posibilități.



Stări S_1	Stări S_2	Linie bus neconectată
0	0	B_3
0	1	B_2
1	1	B_1

c)

Fig. 2.28. Structură redondantă de comutație aplicată bus-urilor de date:
a — schema electrică a comutatoarelor; *b* — celule bistabile pentru memorarea stării bus-ului; *c* — stările bus-urilor în sistemul redondant.

O analiză calitativă a celor două scheme arată că, pentru cea de-a doua schemă, complexitatea structurii, ca și cerința de fan-in/fan-out au crescut de aproximativ trei ori, dar și caracteristicile de fiabilitate s-au îmbunătățit.

2.3.2. APLICAREA STRUCTURII REDONDANTE ÎN CAZUL A l LINII DE INFORMAȚIE

Schemele de mai sus pot fi generalizate pentru cazul a l linii de informație și k rezerve.

Atunci cînd $k > 1$, devine necesar ca registrul BSR să poată evidenția mai mult de un bit. Fie b numărul de circuite basculante bistabile ale fiecărui registru. Dacă BSR trebuie să evidențieze $k + 1$ stări posibile ale liniei de informație, atunci evident:

$$b > \log_2(k + 1). \tag{2.13}$$

Următoarea problemă în proiectarea unui astfel de sistem este stabilirea modulului în care aceste $k + 1$ stări sînt codate, utilizîndu-se cei b biți disponibili ai registrului BSR. Aceasta afectează atît complexitatea, cît și fiabilitatea sistemului.

Atunci cînd numărul rezervelor este mic ($k \leqslant 7$), decodarea este realizată de primul nivel logic al comutatoarelor, fără o creștere a fan-in-ului cerut pentru BSR. În schimb, factorul de fan-out al registrelor BSR poate deveni neacceptabil de mare, dacă l este foarte mare.

O altă soluție constă în asocierea unui grup de linii de informație la un grup de linii de rezervă. Dacă g este un divizor comun al lui l și k, atunci cele l linii de informație se pot divide în g grupe de cîte l/g linii fiecare și pentru fiecare grup se repartizează k/g linii de rezervă ($1 < g \leqslant l$). Mărindu-se valoarea lui g, complexitatea globală se reduce,

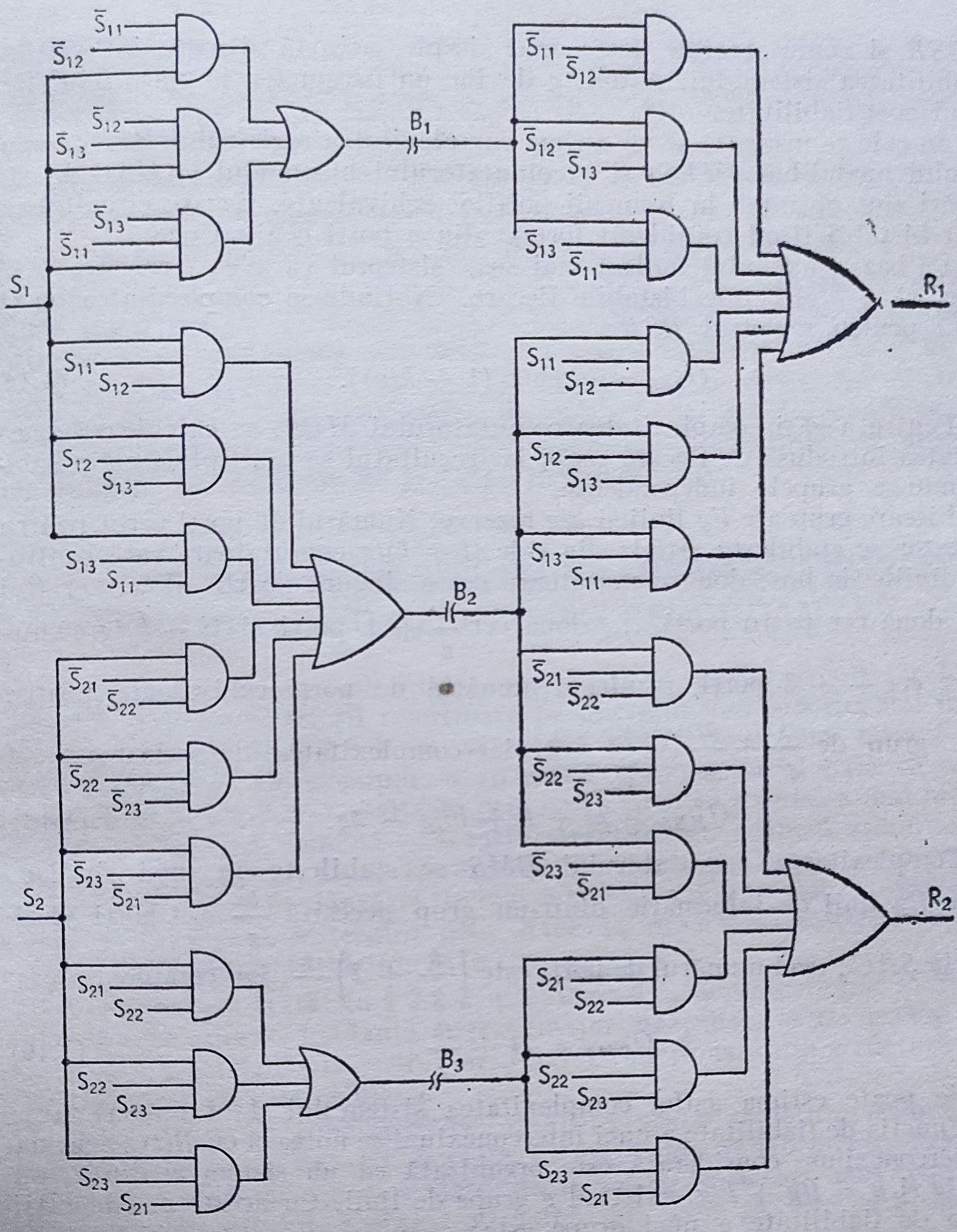


Fig. 2.29. Aplicarea unei structuri redondante pentru registrul stărilor și sistemu-
lui de comutatoare.

iar BSR și comutatoarele devin mai simple, această situație influențînd și fiabilitatea sistemului. Astfel, g devine un parametru important al factorului cost/fiabilitate.

În cele ce urmează se va estima complexitatea registrului BSR, comutatorului modul-bus (MBS) și a comutatorului bus-modul (BMS). Aceste estimări sînt obținute în termenii porților echivalente, fiecare circuit basculant bistabil fiind considerat format din a porți echivalente.

Pe baza modelului indicat mai sus, sistemul va avea l registre BSR cu $\log_2(1 + k/g)$ celule bistabile fiecare. Notîndu-se complexitatea cu Q, rezultă pentru registrul BSR:

$$Q_{BSR} = al\,[\log_2(1 + k/g)]. \tag{2.14}$$

Pentru a găsi complexitatea comutatorului MBS, se calculează complexitatea introdusă de fiecare grup, iar rezultatul se multiplică cu g, considerîndu-se grupele independente.

Fiecare grup are l/g linii și k/g rezerve. Numărul de porți cerut pentru comutator se stabilește astfel: din cele $(l + k)/g$ comutatoare care controlează liniile de bus, două comutatoare cer o singură poartă, două cer trei porți, două cer patru porți, ..., două cer $\frac{k}{g} + 1$ porți și $(l - k)/g$ comutatoare cer $\frac{k}{g} + 2$ porți, rezultînd numărul de porți echivalente pentru fiecare grup de $\frac{lk}{g^2} + \frac{2l}{g} + \frac{k}{g} - 2$, iar complexitatea de valoarea:

$$Q_{MBS} = 2l + k + lk/g - 2g. \tag{2.15}$$

Complexitatea comutatorului BMS se stabilește în mod similar. Fiecare l/g linii de informație dintr-un grup necesită $\frac{k}{g} + 1$ porți $ȘI$ și o poartă SAU; deci numărul de porți este $\left(\frac{k}{g} + 2\right)\frac{l}{g}$, iar complexitatea:

$$Q_{BMS} = 2l + \frac{lk}{g}. \tag{2.16}$$

Se poate estima astfel complexitatea sistemului de interconectare

Funcția de fiabilitate a unei interconexiuni se notează cu $R(t)$. Schema de interconexiune considerată este organizată ca un sistem m din n, cu $m = l/g$ și $n = l/g + k/g$, existînd g grupe de linii. Cu aceste considerații funcția de fiabilitate a unei grupe este:

$$R(t)_{m,n} = \sum_{j=0}^{n-m} C_n^j \, \bar{R}(t)^{n-j} \, (1 - R(t))^j, \tag{2.17}$$

iar pentru cele g grupe ale sistemului de interconectare funcția de fiabilitate are expresia:

$$R(t)_{sist} = \left[\sum_{j=0}^{k/g} C_{l/g+k/g}^j \, R(t)^{l/g+k/g-j} \cdot (1 - R(t))^j \right]^g. \tag{2.18}$$

Ecuațiile (2.14) ÷ (2.18) evidențiază dependența complexității și a funcției de fiabilitate de g, permițînd o optimizare a structurii sistemului.

În concluzie, alegerea numărului de conexiuni de rezervă, k, și a numărului de grupe, g, pentru un număr dat de linii de informație, l, ca și valoarea funcției de fiabilitate pentru o linie de bus, $R(t)$, este influențată de următorii trei factori:

(1) valoarea funcției de fiabilitate pentru sistemul de interconectare;

(2) creșterea în complexitate, care trebuie să fie limitată;

(3) mărirea numărului de conexiuni intrare-ieșire permis.

Valorile k și g se aleg, pe baza ecuației (2.18), care permite stabilirea tuturor perechilor (k, g), astfel încît să fie îndeplinită condiția (1). Dintre aceste valori se aleg acelea care satisfac cel mai bine condițiile (2) și (3).

Alte structuri tolerante la defectări implementate sistemelor de interconectare pot fi de tip „gracefully degrading", structuri reconfigurabile pentru care bus-ul își modifică — micșorîndu-și — capacitatea la apariția unor defectări (de exemplu de la 32 la 16 sau 8 biți).

2.4. PROBLEME DE SINCRONIZARE ÎN SISTEMELE DIGITALE CU STRUCTURĂ REDONDANTĂ

Implementarea unei structuri redondante în cazul circuitelor logice implică o serie de restricții referitoare la sincronizarea funcționării unităților constituente. În § 2.2.1 s-a arătat că aplicîndu-se unui circuit basculant monostabil, de exemplu, o structură redondantă globală ca aceea prezentată în figura 2.5 — realizată prin conectarea în paralel a mai multor circuite pe o sarcină comună — această structură nu acoperă anumite defectări. Inconvenientul poate fi rezolvat prin stabilirea, pentru circuitul dat, unei structuri redondante de tip logică majoritară. În acest caz impulsul va apărea la ieșirea voterului doar dacă la momentul respectiv acesta există la majoritatea intrărilor sale.

După cum s-a arătat în § 2.2.2, probleme de sincronizare pot să apară, și pentru structura redondantă logică majoritară. Este — de exemplu — cazul cînd semnalele de la ieșirea modulelor funcționale din structură sînt întîrziate unele față de celelalte.

Astfel, dacă un generator de impulsuri are o structură redondantă logică majoritară cu $2n - 1$ module identice dar independente, rezultă un sistem cu performanțe mai slabe decît cel cu structură neredondantă. Este practic imposibil de a obține o sincronizare perfectă între cele 2n — 1 semnale ale modulelor componente. Aceasta atrage după sine imperfecțiunea sistemului cu redondanță logică majoritară, de dependență a ieșirii voterului de ieșirile eronate ale modulelor funcționale.

De exemplu, în cazul structurii redondante cu logică majoritară de tip 2 din 3, nesincronizarea ca și întîrzierile fizice diferite introduse de către cele trei module funcționale atrage inperfecțiunea funcției logică majoritară pe intervalul de timp $t_2 - t_1$, respectiv $t_4 - t_3$ (fig. 2.30). În fig. 2.30 X_1, X_2, X_3 reprezintă semnalele generate de cele trei module funcționale ale structurii, iar Y — semnalul de la ieșirea sistemului de decizie. Pentru duratele de timp $t_2 - t_1$ și $t_4 - t_3$, semnalul Y depinde de semnalul eronat (defazat) — X_3.

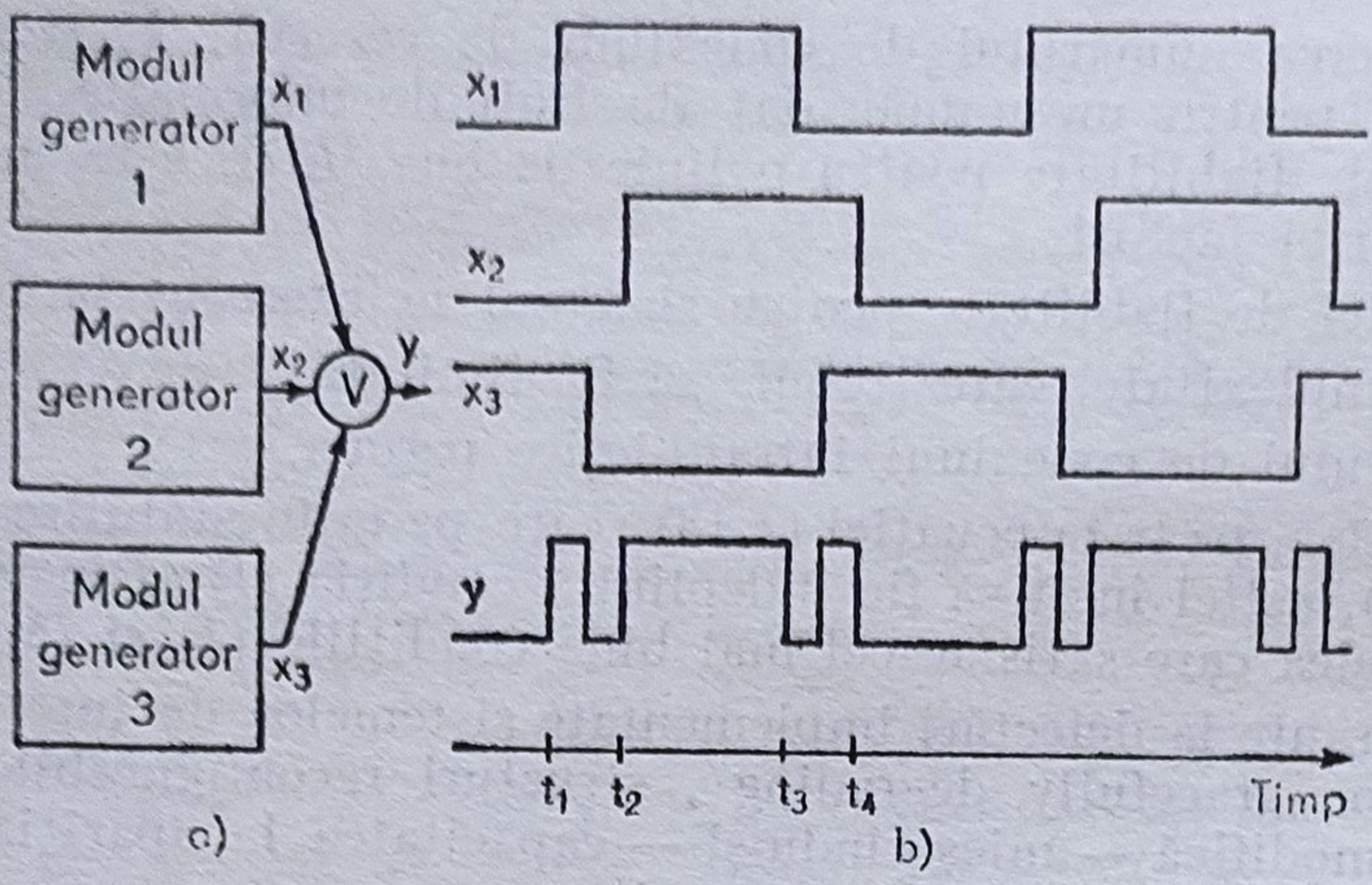


Fig. 2.30. Evidențierea unei imperfecțiuni a funcției logică majoritară:
a — generator de impulsuri cu structură redondantă logică majoritară de tip 2 din 3; *b* — semnalul de ieșire eronat.

Problema se poate generaliza în cazul unei structuri redondante de tip logică majoritară n din $2n - 1$. În acest caz intervalul de timp de insuficiență a funcției logică majoritară este:

$$\Delta t_{max} = \max(t_{\Delta M1M2}, \ldots, t_{\Delta M1M_n}, \ldots, t_{\Delta M_iM_j}, \ldots, t_{\Delta M_{n-2}M_{2n-1}}), \quad (2.19)$$

unde $t_{\Delta M_iM_j}$ este întîrzierea tranzițiilor modulelor generatoare M_i și M_j.

Dacă răspunsul voterului este dat cu o întîrziere t_D după prima tranziție și dacă t_D satisface inegalitățile:

$$t_{\Delta max} < t_D < T/2 - t_{\Delta max}, \quad (2.20)$$

T fiind perioada semnalului generat de generator, neajunsul enunțat al funcției logică majoritară este eliminat.

Pe baza acestui considerent este conceput generatorul de impulsuri cu o structură redondantă logică majoritară de tip 2 din 3, a cărui schemă electrică este indicată în figura 2.31.

Se fac următoarele precizări [4]:

(a) s-a implementat o structură redondantă logică majoritară multiplă pentru cazul $2n - 1 = 3$;

(b) întîrzierea t_D definită de relația (2.20) este realizată de circuitul basculant monostabil $MONO_1$, al cărui semnal de ieșire formează semnalul de tact pentru circuitul basculant bistabil, B;

(c) oscilația este obținută în bucla formată de voter și circuitul basculant bistabil B, care memorează complementul funcției majoritare la fiecare tranziție a semnalului de tact generat;

(d) CI este un circuit de întîrziere care realizează întîrzierea semnalului pe bucla de reacție cu timpul τ. Astfel, semiperioada semnalului generat este:

$$T/2 = t_D + \tau.$$

De aici se obține, în conformitate cu (2.20), relația de proiectare pentru mărimea de timp τ.

$$\tau > t_{\Delta max}.$$

Fig. 2.31. Schema electrică a unui generator de impulsuri cu structură redondantă logică majoritară cu sincronizarea semnalelor generate.

Timpul τ este generat de circuitul monostabil $MONO_2$.

De subliniat că asigurarea sincronizării funcționării modulelor multiplicate ale unei structuri redondante aplicate sistemelor digitale este o problemă importantă în implementarea toleranței la defectări pentru aceste sisteme. În general, această sincronizare este asigurată de un semnal de sincronizare extern, care determină momentele de votare sau validare a informației de la modulele active. Generatorul de impulsuri prezentat mai sus asigură pe calea de reacție o votare sincronă a semnalelor generate de cele trei module funcționale din structură.

2.5. CRITERII DE COMPARARE A STRUCTURILOR REDONDANTE

O metodă care face posibilă compararea diferitelor structuri redondante ce pot fi aplicate unui sistem dat presupune evidențierea atît a îmbunătățirii fiabilității sistemului prin utilizarea redondanței, cît și a creșterii costului acestuia ca urmare a introducerii unor elemente suplimentare în sistem; valorile acestor două determinări caracterizează performanțele sistemului cu structură redondantă și servesc ca bază de comparație.

În continuare, se prezintă o metodă de comparare a structurilor redondante din punctul de vedere al performanțelor, dezvoltată pe baza lucrărilor [5], [28], [34]. Se propune utilizarea, în acest scop, a indicelui logaritmic de îmbunătățire a fiabilității, a indicilor costului și eficienței pentru sistemul cu structură redondantă în raport cu cel cu structură neredondantă.

Se au în vedere următoarele ipoteze:

— defectările componentelor sistemului sînt considerate statistic independente;

— în scopul facilității studiului comparativ, se consideră că subsistemelor cu structură neredondantă din sistemul inițial le corespund același număr de subsisteme cu structură redondantă.

Pentru dezvoltările de mai jos, se consideră că sistemul cu structură neredondantă este format din k subsisteme identice din punctul de vedere al fiabilității, iar sistemul cu structură redondantă — din același număr de subsisteme identice, dar cu structură *redondantă*, astfel încît expresiile funcțiilor de fiabilitate pentru sistemul neredondant, R_{SN}, respectiv pentru sistemul redondant, R_{SR}, sînt:

$$R_{SN}(t) = R_n(t)^k; \qquad (2.21)$$

$$R_{SR}(t) = R_r(t)^k$$

În cele ce urmează în afara notațiilor generale se utilizează următoarele notații: k — numărul subsistemelor identice din punct de vedere al fiabilității; I — indicii de îmbunătățire a fiabilității; I_c — indicele costului; E — indicatorul global al eficienței.

Pentru facilitarea scrierii mărimilor dependente de timp — atunci cînd nu se urmărește evidențierea în mod explicit a dependenței de timp a mărimilor respective — acestea vor fi notate simplificat, fără a se mai scrie argumentul timp. De exemplu, $R(t)$ va fi notat cu R sau $F(t)$, cu F.

2.5.1. INDICI DE ÎMBUNĂTĂȚIRE A FIABILITĂȚII SISTEMELOR CU STRUCTURĂ REDONDATĂ

Pentru a permite evaluarea îmbunătățirii fiabilității unui sistem prin aplicarea redondanței, se compară — de obicei — valorile numerice ale unui indice de creștere a fiabilității. Un astfel de indice trebuie:

(a) să fie independent de imărimea sistemului căruia i se aplică redondanța;

(b) să fie sensibil la schimbări mici în valoarea funcției de fiabilitate, în special cînd aceasta este apropiată de 1.

În continuare vor fi analizate mai multe exprimări posibile ale indicelui de creștere a fiabilității:

• Indicele de îmbunătățire a fiabilității sistemului, exprimat ca raportul probabilităților de bună funcționare pentru sistemul cu structură redondantă și neredondantă, definit cu relația:

$$I_R = \frac{R_{SR}}{R_{SN}} = \left(\frac{R_r}{R_n}\right)^k, \tag{2.22}$$

dedusă ținîndu-se seama de expresia (2.21);

• Indicele de îmbunătățire a fiabilității sistemului, exprimat ca raport al probabilităților de defectare pentru sistemele cu structură neredondantă, și, respectiv cu structură redondantă, definit cu relația:

$$I_F = \frac{F_{SN}}{F_{SR}} = \frac{1-R_{SN}}{1-R_{SR}} = \frac{1-R_n^k}{1-R_r^k}. \tag{2.23}$$

Deoarece în expresiile lui I_R și I_F apare k, înseamnă că indicii respectivi sînt dependenți de mărimea sistemului, deci contravin condiției (a) enunțată mai sus;

• Indicele de îmbunătățire a fiabilității sistemului, exprimat ca raportul timpilor medii de bună funcționare pentru sistemele cu structură redondantă și, respectiv, cu structură neredondantă, definit cu relația:

$$I_m = \frac{m_{SR}}{m_{SN}}, \tag{2.24}$$

unde m este media timpului de bună funcționare.

Deoarece

$$m = \int_0^\infty -\frac{\mathrm{d}R}{\mathrm{d}t}\, t\, \mathrm{d}t,$$

expresia (2.24) dată indicelui I_m devine:

$$I_m = \frac{\int_0^\infty \frac{\mathrm{d}R_{SR}}{\mathrm{d}t}\, t\, \mathrm{d}t}{\int_0^\infty \frac{\mathrm{d}R_{SN}}{\mathrm{d}t}\, t\, \mathrm{d}t}. \tag{2.25}$$

În condițiile ipotezelor inițiale și ținîndu-se seama de relația (2.21) rezultă:

$$I_m = \frac{\int_0^\infty R_r^{k-1} \cdot \frac{dR_r}{dt}\, dt}{\int_0^\infty R_n^{k-1} \cdot \frac{dR_n}{dt}\, dt}. \tag{2.26}$$

Expresia lui I_m conține numărul k, deci indicele I_m este dependent de mărimea sistemului, ceea ce contravine condiției (a);

• Indicele de îmbunătățire a fiabilității sistemului, exprimat ca raportul ratelor de defectare pentru sistemele cu structură neredondantă, Z_{SN}, și redondantă, Z_{SR}, definit cu relația:

$$I_z = \frac{z_{SN}(t)}{z_{SR}(t)}. \tag{2.27}$$

Deoarece $z(t) = -\frac{1}{R(t)} \frac{dR(t)}{dt}$, ecuația 2.27) devine:

$$I_z = \frac{R_{SR} \cdot \frac{dR_{SN}}{dt}}{R_{SN} \cdot \frac{dR_{SR}}{dt}}. \tag{2.28}$$

În condițiile ipotezelor (2.21), ecuația (2.28) devine:

$$I_z = \frac{R_r \frac{dR_n}{dt}}{R_n \frac{dR_r}{dt}}. \tag{2.29}$$

Expresia (2.29) obținută pentru indicele I_z este independentă de dimensiunea sistemului, dar indicele I_z nu este sensibil la variațiile lui R_r în jurul lui 1.

Indicele logaritmic de îmbunătățire a fiabilității sistemului cu structură redondantă este definit cu relația:

$$I_{\log} = \alpha = \frac{\ln R_{SN}(t)}{\ln R_{SR}(t)}. \tag{2.30}$$

Avînd în vedere ipotezele (2.21), expresia indicelui $I_{\log}$ dat de ecuația (2.30) devine:

$$I_{\log} = \alpha = \frac{\ln R_n(t)}{\ln_r R(t)}. \tag{2.31}$$

Dezvoltîndu-se în serie logaritmul neperian, rezultă:

$$\alpha = \frac{\ln R_n}{\ln R_r} = \frac{\ln (1 - F_n)}{\ln (1 - F_r)} = \frac{F_n + F_r^2/2 + \dots}{F_r + F_r^2/2 + \dots};$$

pentru F_n, $F_r \ll 1$ se poate face aproximația:

$$\alpha = \frac{F_n (1 + F_n/2 + \dots)}{F_r (1 + F_r/2 + \dots)} \simeq \frac{F_n}{F_r} \left(1 + \frac{F_n - F_r}{2}\right). \tag{2.32}$$

Așa cum va rezulta din exemplele numerice ale studiilor de caz de mai jos, pentru valorile uzuale ale funcțiilor de repartiție ale timpului de bună funcționare, eroarea determinată prin utilizarea relației aproximative (2.32) este neglijabilă.

Față de toți ceilalți indici de îmbunătățire a fiabilității sistemului cu structură redondantă prezentați mai sus, indicele α este independent de mărimea sistemului și este sensibil la variații mici în valoarea funcției de fiabilitate pentru valori ale acesteia apropiate de 1, ceea ce recomandă folosirea sa în studiile privind creșterea fiabilității sistemelor prin utilizarea redondanței. În continuare indicele α va fi utilizat în acest scop în vederea comparării diferitelor tehnici de redondanță aplicate sistemelor digitale.

2.5.2. INDICI DE COST ȘI EFICIENȚĂ

Aplicarea redondanței este adeseori limitată de considerente tehnologice și economice. Pentru evaluarea acestor limitări, devine necesară introducerea indicilor de cost. O exprimare posibilă a unui astfel de indice, este:

$$I_c = \frac{\text{costul sistemului redondant}}{\text{costul sistemului neredondant}}. \qquad (2.33)$$

Noțiunea de cost este privită aici în sens larg, ca un parametru sau o combinație de parametri a căror creștere evidențiază dezavantajul măririi gradului redondanței.

Parametrii costului — greutate, volum, manoperă etc. — sînt dependenți, dacă se consideră cazul circuitelor electronice, de: numărul componentelor, al interconexiunilor, al porților logice din capsula circuitului integrat ș.a.

Pentru circuitele logice se poate folosi fie indicele de cost care ia în considerație numărul porților pe capsulă, I_{cp}, definit cu relația:

$$I_{cp} = \frac{\text{numărul de porți ale sistemului redondant}}{\text{numărul de porți ale sistemului neredondant}}, \qquad (2.34)$$

fie indicele de cost care ia în considerație numărul de intrări în poartă (poartă echivalentă), I_{ci}, definit cu relația:

$$I_{ci} = \frac{\text{numărul de intrări în poartă pentru sistemul redondant}}{\text{numărul de intrări în poartă pentru sistemul neredondant}}. \qquad (2.35)$$

Indicele I_{cp} este util atunci cînd se consideră costul global al sistemului, greutatea sau volumul în cazul sistemelor construite cu circuite integrate pe scară redusă, sau consumul de putere pentru cazul utilizării oricărei tehnologii de realizare a circuitelor. Indicele I_{ci} este reprezentativ atunci cînd se consideră costul, greutatea sau volumul sistemului construit cu circuite integrate pe scară largă.

În scopul evaluării eficienței utilizării tehnicilor de redondanță se poate utiliza indicele de eficiență, E, definit [5] cu relația:

$$E = \frac{\text{indicele logaritmic de creștere a fiabilității}}{\text{indicele de cost}} \tag{2.36}$$

unde indicele de cost poate fi I_{cp} sau I_{ci}.

Studiile de caz prezentate în continuare vor evidenția utilitatea indicilor definiți anterior în cazul sistemelor electronice cu structură redondantă.

2.5.3. STUDII DE CAZ PRIVIND EVALUAREA PERFORMANȚELOR UNOR SISTEME CU STRUCTURĂ REDONDANTĂ

În cele ce urmează se vor evalua performanțele a trei structuri redondante utilizabile în implementarea sistemelor tolerante la defectări, folosind — în vederea comparării celor trei tipuri de structuri — indicii definiți mai sus. În termenii metodei propuse, se caută de fiecare dată un *nivel optim* de aplicare a tehnicii redondante respective în sistem.

Ipoteze și notații

• Pentru evaluarea funcției de fiabilitate a sistemului cu structură redondantă se vor utiliza atît ecuațiile deduse din modelul structural de evaluare a fiabilității sistemelor [14], cît și ecuațiile unui model bazat pe metoda tăieturilor minimale, [8], care ține seama de modurile de defectare a sistemului, numit în cele ce urmează modelul *modurilor de defectare*.

• Se consideră că defectările elementelor sistemului sînt statistic independente.

• Se presupune că pentru circuitele logice există două tipuri de semnale false (moduri de defectare): semnal logic binar „zero fals", notat $\varnothing$, și semnal logic binar „unu fals", notat 1, care pot să apară cu probabilități egale, adică

$$F_{\varnothing} = F_{1} = \frac{F}{2}. \tag{2.37}$$

• Analizele care urmează se fac pe scheme cu porți logice în tehnologie TTL. Se iau în considerație doar două moduri distincte de defectare ale unui tranzistor dintr-o poartă logică: scurtcircuit și întreruperi; cele două moduri de defectare se presupun echiprobabile:

$$F_{TO} = F_{TS} = F_T/2 \tag{2.38}$$

• Defectările restului componentelor din poartă se consideră că apar cu probabilități neglijabile.

A. Determinarea indicilor de creștere a fiabilității, de cost și eficiență pentru structura redondantă logică majoritară simplă

Structura redondantă logică majoritară se poate aplica oricăror elemente funcționale ale unui sistem; interconexiunile între modulele funcționale neredondante ale acestei structuri nu sînt redondante, iar modulele funcțio-

nale se conectează în sistemul redondant în același mod ca în cazul sistemului neredondant (v. § 2.2.2.). Voterul poate fi sintetizat în mai multe moduri; în figura 2.32 este prezentat un astfel de circuit realizat cu porți NAND.

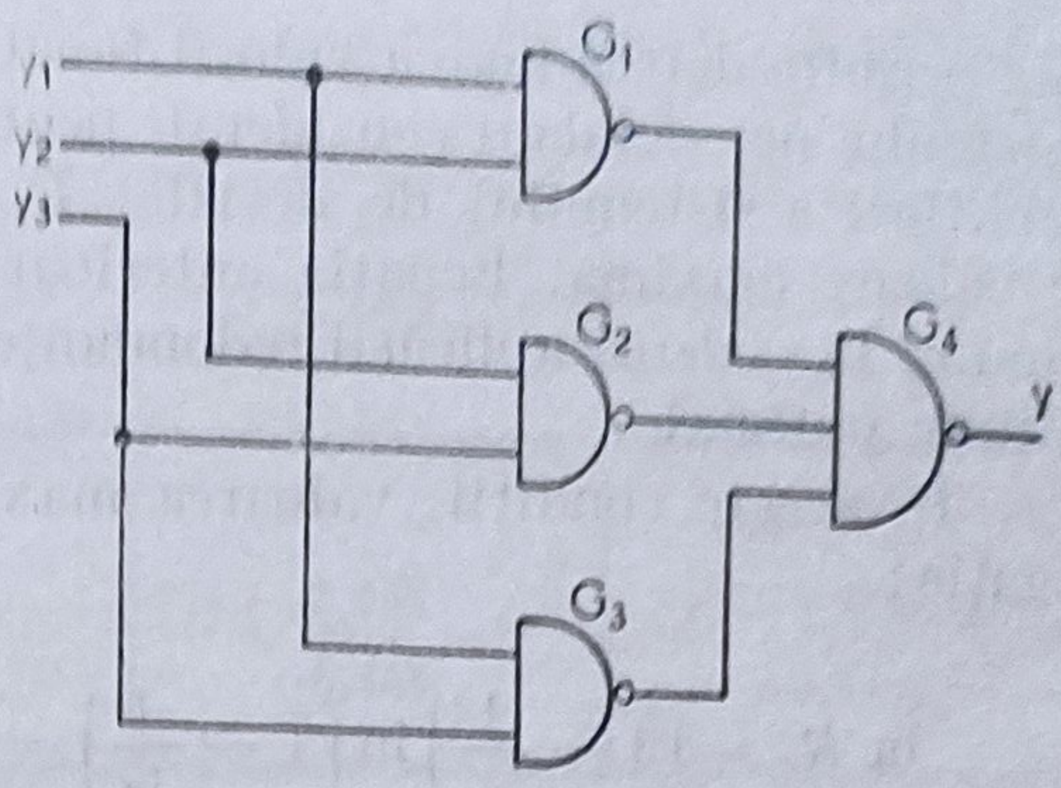


Fig. 2.32. Realizarea unui voter cu porți NAND

Modelul uzual de evaluare a bunei funcționări a unui sistem cu structură redondantă cu logică majoritată, adoptat în majoritatea lucrărilor de specialitate, presupune că atunci cînd are loc un defect într-unul din cele trei module funcționale ale structurii, iar celelalte două funcționează corect, ieșirea sistemului de decizie va fi corectă, cu condiția ca însuși acest sistem să fie în stare de bună funcționare.

Analiza funcționării unei astfel de structuri, utilizată în cazul circuitelor logice, arată că, dacă la una dintre ieșirile celor trei module ale structurii apare semnal logic *1*, iar la o altă ieșire apare semnalul logic *0*, structura redondantă va masca ambele tipuri de defectări; dacă însă cele două module prezintă defectări de același tip, acestea nu mai pot fi mascate la ieșirea sistemului.

O analiză a funcționării circuitului care formează sistemul de decizie arată că, atunci cînd intrările sale sînt identice, acesta poate tolera anumite tipuri de defectări; de exemplu, o blocare în „1" a porții G_1 este msacată cu condiția ca semnalele de intrare, Y_1, Y_2 și Y_3, să fie identice. Această redondanță funcțională a circuitului de decizie trebuie să fie luată în considerație pentru evaluarea fiabilității întregului sistem, mai ales pentru o comparație realistă a diferitelor tehnici de redondanță.

A.1. *Determinarea indicilor de creștere a fiabilității, de cost și eficiență cu modelul uzual de evaluare a indicatorilor de fiabilitate*

• Funcția de fiabilitate se modelează cu ecuația (2.3) aplicată pentru cazul redondanței de tip 2 din 3,

$$R_{s\,R} = (3R_n^2 - 2R_n^3)R_V. \tag{2.39}$$

• În aceste condiții indicele de creștere a fiabilității, determinat conform expresiei (2.31), va fi:

$$\alpha = \frac{\ln R_n}{\ln \left[\left(3R_n^2 - 2R_n^3\right) R_V\right]}. \tag{2.40}$$

Prin derivarea funcției α în raport cu R_n și egalarea cu zero a derivatei, se obține ecuația:

$$\ln R_V = -\frac{2R_n}{3-2R_n}\ln R_n - \ln(3-2R_n), \tag{2.41}$$

care permite determinarea valorii funcției de fiabilitate, R_n, ce se cere subsistemului neredondant considerat, pentru o anumită valoare a funcției de fiabilitate a sistemului de decizie, R_V, astfel încît să se obțină un indice α de valoare maximă. Ecuația anterioară dă informații asupra partiției sistemului în vederea aplicării redondanței la nivelul unor performanțe de fiabilitate optimale.

În aceste condiții, valoarea maximă a indicelui α se obține [5] din ecuația:

$$\ln R_r = \left(3 - \frac{1}{\alpha}\right) \ln\left(1 - \frac{1}{3\alpha}\right) - \left(2 - \frac{1}{\alpha}\right) \ln\left(1 - \frac{1}{2\alpha}\right) - \ln\alpha. \quad (2.42)$$

Ecuațiile (2.41) și (2.42) definesc condițiile de implementare a unei structuri redondante logică majoritară de tip 2 din 3, astfel încît indicele α să fie de valoare maximă.

• Pe baza conceptului tăieturilor minimale se întocmește tabelul 2.5, care indică tăieturile minimale ale structurii redondante cu logică majoritară de tip 2 din 3, de unde se deduce ușor că, în domeniul valorilor maximale ale funcției de fiabilitate, expresia funcției de repartiție a timpului de funcționare fără defectări pentru structura considerată este:

$$F_r = F_v + 3F_n^2, \quad (2.43)$$

iar indicele de creștere a fiabilității determinat cu (2.32) are expresia:

$$\alpha \simeq \frac{F_n}{F_v + 3F_n^2}. \quad (2,44)$$

Eroarea rezultată atunci cînd se utilizează formula aproximativă (2.44) în locul celei exacte (2.40) este calculată în tabelul 2.6.

Se observă că pentru valorile uzuale ale lui F_n (valori mici) eroarea ce apare datorită utilizării relației (2.44) pentru determinarea indicelui α este mai mică decît 1%, deci poate fi considerată neglijabilă.

Tabelul 2.5. MULȚIMEA TĂIETURILOR MINIMALE PENTRU STRUCTURA REDONDANTĂ LOGICĂ MAJORITARĂ 2 DIN 3

Nr. de ordine al tăieturii minimale	Elemente defecte				Probabilitatea apariției tăieturii minimale
	Module funcționale			Voter	
	1	2	3		
1	×	×			F_n^2
2		×	×		F_n^2
3	×		×		F_v^2
4				×	F_v

Tabelul 2.6. VALORI NUMERICE ALE INDICELUI LOGARITMIC DE CREȘTERE A FIABILITĂȚII

F_n	F_v	α aproximativ	α exact	$\Delta\alpha$
0,2	0,1	0,909	1,036	−12,36%
0,05	0,01	2,857	2,956	−3,35%
0,02	0,01	9,091	9,168	−0,84%

Pe baza ecuației (2.44) se determină valoarea optimă a funcției F_n, care maximizează indicele α:

$$F_{n\,\mathrm{opt}} = \sqrt{\frac{F_v}{3}}\,, \tag{2.45}$$

respectiv

$$\alpha_{\max} = \frac{1}{2\sqrt{3F_v}}\,. \tag{2.46}$$

Deoarece atunci cînd se implementează o astfel de structură numărul modulelor funcționale se multiplică cu trei, iar adăugarea voterului pentru decizie crește complexitatea sistemului cu raportul F_v/F_n, indicele costului se poate calcula cu relația:

$$I_c = 3 + \frac{F_v}{F_n}\,. \tag{2.47}$$

Implementîndu-se structura redondantă la nivel optimal — ecuația (2.45) — , rezultă:

$$I_{c\,\mathrm{opt}} = 3 + \sqrt{3F_v}\,. \tag{2.48}$$

Indicele eficiență definit cu ecuația (2.36) rezultă din expresia:

$$E = \frac{F_n}{F_v + 3F_n^2} \Big/ \left(3 + \frac{F_v}{F_n}\right) \simeq \frac{F_n}{3\left(F_v + 3F_n^2\right)}\,. \tag{2.49}$$

A 2. *Determinarea indicilor de creștere a fiabilității, de cost și eficiență utilizîndu-se modelul modurilor de defectare pentru evaluarea indicatorilor de fiabilitate*

S-a arătat că structura redondantă analizată admite mai multe tipuri de defectări decît cele acceptate de modelul structural de fiabilitate, din care cauză rezultatele obținute în determinarea indicilor de creștere a fiabilității, de cost și eficiență utilizînd modelul structural de fiabilitate sînt mai pesimiste decît dacă s-ar utiliza modelul *modurilor de defectare*. Aceasta

din urmă conduce la rezultate care permit o comparație mai realistă cu alte tipuri de redondanțe, cu toate că există dezavantajul că rezultatele analizei nu pot fi extrapolate, ele fiind valabile doar pentru fiecare schemă particulară.

• Se consideră structura redondantă cu logică majoritară de tip 2 din 3, avînd circuitul sistemului de decizie cu structura indicată în figura 2.32. În vederea evaluării funcției de repartiție a timpului de funcționare se utilizează metoda tăieturilor minimale, avîndu-se în vedere modurile de defectare ale acestei scheme. În tabelul 2.7 se indică mulțimea tăieturilor minimale pentru structura redondantă considerată.

O b s e r v a ț i e. După cum rezultă din [7], ponderea cea mai mare a defectărilor circuitelor integrate bipolare de mică complexitate — cazul exemplului considerat — este repartizată interconexiunilor și zonelor active. De aceea, în tabelul 2.7 s-a considerat că principalele defectări (tăieturi minimale) ale porților logice din componența voterului sînt defectările tranzistoarelor echivalente T_1, T_2 sau T_3, corespunzătoare — în principal — — defectelor joncțiunilor J_1, J_2 sau J_3 ale tranzistorului multiemitor de la intrarea porții *NAND* (fig. 2.32).

• Se poate scrie, dispunînd de mulțimea tăieturilor minimale, probabilitatea de defectare a sistemului cu structură redondantă analizat:

$$F_r = 3F_{TS} + 3F_{n\varnothing}^2 + 3F_{n1}^2. \tag{2.50}$$

În condițiile ipotezelor exprimate de egalitățile (2.37) și (2.38), această ecuație devine:

$$F_r = \frac{3}{2}(F_T + F_n^2). \tag{2.51}$$

O b s e r v a ț i e. Din relația (2.51) se observă că circuitul voterului contribuie la valoarea probabilității de defectare cu $1{,}5\,F_T$, față de $3\,F_T$ cît ar fi fost contribuția acestuia dacă se evalua probabilitatea de defectare cu modelul structural de fiabilitate. Se evidențiază astfel caracterul mai pesimist al evaluărilor de fiabilitate folosindu-se modelul structural.

• Indicele α de creștere a fiabilității are expresia:

$$\alpha = \frac{2F_n}{3(F_T + F_n^2)}. \tag{2.52}$$

Valoarea funcției de repartiție a timpului de funcționare pentru unitatea neredondantă, F_n, care maximizează indicele α este:

$$F_{n\,\mathrm{opt}} = \sqrt{F_T}, \tag{2.53}$$

iar α_{max} va lua valoarea:

$$\alpha_{max} = \frac{1}{3\sqrt{F_T}}. \tag{2.54}$$

Tabelul 2.7. TĂIETURI MINIMALE PENTRU STRUCTURA REDONDANTĂ LOGICĂ MAJORITARĂ 2 DIN 3

Nr. de ordine al tăieturii minimale	Modul funcțional/ poartă	M_1		M_2		M_3		G_1				G_2				G_3				G_4						Probabilitatea de apariție
	Joncțiune tranzistor							J_1		J_2		J_1		J_2		J_1		J_2		J_1		J_2		J_3		
	Mod de defectare	0	1	0	1	0	1	S	G	S	G	S	G	S	G	S	G	S	G	S	G	S	G	S	G	
1																				×						$3\,F_{TS}$
2																						×				
3																								×		
4		×		×																						$3\,F^2_{n0}$
5		×				×																				
6				×		×																				
7			×		×																					$3\,F^2_{n1}$
8			×				×																			
9					×		×																			
10									×		×															$3\,F^2_{T0}$
11													×		×											
12																	×		×							

O b s e r v a ț i e. Desigur, puteau fi luate în considerație mai multe mulțimi de defectări, în care caz valorile indicate de relațiile (2.50) ÷ (2.54) se obțineau cu acuratețe mai mare; dar pentru o primă analiză realizată în vederea comparării mai multor structuri redondante, aceste expresii oferă suficientă informație.

• Deoarece în această structură redondantă modulele neredondante sînt triplate — într-un modul neredondant sînt un număr de $\frac{F_n}{n_T F_t}$ porți echivalente, iar circuitul voterului conține patru porți — indicele de cost I_{cp} va fi dat de relația:

$$I_{cp} = 3 + 4\, n_T \cdot F_T/F_n. \tag{2.55}$$

• Indicele I_{ci} se obține considerîndu-se că în circuitul voterului sînt nouă intrări și echivalentul de F_n/F_T intrări în porțile modulului neredondant analizat:

$$I_{ci} = 3 + 9\, F_T/F_n. \tag{2.56}$$

• Pentru o structură redondantă aplicată la nivelul optimal stabilit anterior, acești indici vor avea forma:

$$I_{cp\ opt} = 3 + 4\, n_T \sqrt{F_T}; \tag{2.57}$$

$$I_{ci\ opt} = 3 + 9 \sqrt[4]{F_T}. \tag{2.58}$$

• Atunci cînd se ia în considerație mărimea numărului de porți ale sistemului datorită aplicării redondanței, indicele eficiență, E, are expresia:

$$E_p = \frac{2\, F_n}{3\,(F_T + F_n^2)} \Big/ \left(3 + \frac{4\, n_T F_T}{F_n}\right) = \frac{2\, F_n^2}{3\,(F_T + F_n^2)\,(3\, F_n + 4\, n_T F_T)}. \tag{2.59}$$

Dacă se ține seama de creșterea numărului de intrări în sistem prin aplicarea redondanței, indicele eficiență, E, devine:

$$E_i = \frac{2\, F_n}{3\,(F_T + F_n^2)} \Big/ (3 + 9\, F_T/F_n) = \frac{2\, F_n^2}{3\,(F_T + F_n^2)\,(3\, F_n + 9\, F_T)}. \tag{2.60}$$

A.3. *Rezultate numerice*

Structura redondantă logică majoritară poate fi aplicată unui sistem digital la orice nivel al componenței sale. S-a determinat anterior un nivel optim de aplicare a acestei structuri redondante, pentru care se obține indicele α de valoare maximă.

1°. *Redondanța la nivel optim*. Considerîndu-se $F_v = 9F_T$, $n_T = 3$ în ecuațiile (2.45), (2.46) și (2.48), se obțin pentru α_{max}, $I_{c\,opt}$, $F_{n\,opt}$ și E_{opt} valorile numerice cuprinse în tabelul 2.8.

Abordîndu-se modelul modurilor de defectare în evaluarea expresiei funcției de repartiție a timpului de funcționare, se obțin pentru α_{max}, $I_{c\,opt}$, $F_{n\,opt}$ și E_{opt} — ecuațiile (2.50), (2.51), (2.54) și (2.55) — valorile numerice cuprinse în tabelul 2.9.

Tabelul 2.8. PERFORMANȚELE OPTIMALE ALE STRUCTURII REDONDANTE LOGICĂ MAJORITARĂ DE TIP 2 DIN 3 CALCULATE CU MODELUL UZUAL DE EVALUARE A FIABILITĂȚII

F_T	10^{-6}	10^{-5}	10^{-4}	10^{-3}
α_{max}	96,2	30,4	9,62	3,04
$I_{c\ opt}$	3,01	3,02	3,05	3,16
$F_{n\ opt}$	0,00173	0,00548	0,0173	0,0548
$E_{o\ opt}$	31,96	10,06	3,15	0,962

Tabelul 2.9. PERFORMANȚELE OPTIMALE ALE STRUCTURII REDONDANTE LOGICĂ MAJORITARĂ DE TIP 2 DIN 3 CALCULATE CU MODELUL MODURILOR DE DEFECTARE

F_T	10^{-6}	10^{-5}	10^{-4}	10^{-3}
α_{max}	333	105	33,3	10,5
$I_{cp\ opt}$	3,01	3,04	3,12	3,38
$I_{ci\ opt}$	3,01	3,03	3,09	3,28
$F_{n\ opt}$	0,001	0,00316	0,01	0,0316
$E_{p\ opt}$	110,6	84,33	10,67	3,106
$E_{i\ opt}$	110,6	34,65	10,77	3,201

• Din analiza acestor rezultate numerice se obțin indicații cu privire la partiționarea sistemului în vederea aplicării structurii redondante la nivel optimal. Astfel, numărul tranzistoarelor echivalente ale unui modul funcțional neredondant, care dă dimensiunea nivelului optimal, variază de la 1 000 la 30, atunci cînd F_T variază de la 10^{-6} la 10^{-3} (tabelul 2.9).

• Rezultatele numerice obținute arată modificări nesemnificative ale indicilor costului pentru diferite valori ale lui F_T și implicit ale lui $F_{n\ opt}$.

2°. *Redondanța aplicată la nivel neoptimal.* Nu întotdeauna este posibilă partiționarea sistemului la dimensiunile optimale (în conformitate cu criteriul „indicele α de creștere a fiabilității este maxim") în vederea aplicării redondanței. De aceea, este important de investigat cum se modifică performanțele schemelor redondante atunci cînd redondanța se aplică la nivel neoptimal.

Rezultatele numerice obținute pe baza ecuațiilor (2.44), (2.47), (2.49), (2.52), (2.53) și (2.55) sînt date în tabelele 2.10 și 2.11 — pentru trei valori ale lui F_T, în ipoteza $F_v = 9F_p$ și $n_T = 3$.

Se observă că indicele costului nu se modifică prea mult la variații mari reale indicelui α de creștere a fiabilității. Această observație sugerează că redondanța logică majoritară aplicată la nivel optimal este eficientă — in-

Tabelul 2.10. PERFORMANȚE ALE STRUCTURII REDONDANTE LOGICĂ MAJORITARĂ DE TIP 2 DIN 3 CALCULATE CU MODELUL UZUAL DE EVALUARE A FIABILITĂȚII

	$FT = 10^{-5}$					$FT = 10^{-4}$				$FT = 10^{-3}$		
Fn	10^{-4}	10^{-3}	5.10^{-3}	10^{-2}	10^{-1}	10^{-3}	10^{-2}	3.10^{-2}	10^{-1}	10^{-2}	5.10^{-2}	10^{-1}
α	1,11	10,8	30,4	25,6	3,3	1,11	8,33	8,62	3,24	1,07	3,04	2,56
I_c	3,9	3,09	3,02	3,01	3	3,9	3,009	3,05	3,01	3,9	3,16	3,09
E	0,284	3,495	10,06	8,5	1,1	0,284	2,695	2,826	1,076	0,274	0,962	0,828

Tabelul 2.11. PERFORMANȚE ALE STRUCTURII REDONDANTE LOGICĂ MAJORITARĂ DE TIP 2 DIN 3 CALCULATE CU MODELUL MODURILOR DE DEFECTARE

	$FT = 10^{-5}$					$FT = 10^{-4}$			$FT = 10^{-3}$		
Fn	10^{-4}	10^{-3}	3.10^{-2}	10^{-2}	10^{-1}	10^{-3}	10^{-2}	10^{-1}	10^{-2}	3.10^{-2}	10^{-1}
α	6,66	60,6	105	60,6	6,66	6,6	33,3	6,6	6,06	10,5	6,06
I_{cp}	4,2	3,12	3,04	3,01	3	4,2	3,12	3,01	4,2	3,38	3,12
I_{ci}	3,9	3,09	3,03	3,01	3	3,9	3,09	3,01	3,9	3,28	3,09
E_p	1,585	19,42	34,54	20,132	2,22	1,57	10,673	2,192	1,442	3,106	1,94
E_i	1,707	19,61	34,65	20,132	2,22	1,69	10,776	2,192	1,553	3,201	1,96

dicele de creștere a fiabilității este maxim, iar costul nu este mai mare decît cel pentru alte niveluri de aplicare a redondanței. Dacă se urmărește un cost mai mic de implementare a structurii redondante, din analiza acestor date numerice se observă că este mai bine să se utilizeze un modul funcțional de dimensiuni mai mari față de optim (F_n mai mare) decît module de dimensiuni mai mici; aceasta datorită faptului că indicele costului este mai mic în primul caz decît în cel de-al doilea.

3°. În continuare — pentru o analiză mai sugestivă — se vor reprezenta grafic rezultatele obținute anterior.

Astfel, în figura 2.33 este reprezentată variația indicelui α funcție de valorile lui F_n, utilizîndu-se modelul modurilor de defectare pentru evaluarea indicatorului de fiabilitate, F_n. Din analiza acestei reprezentări grafice se observă cum creșterea fiabilității este influențată de orice modificare a nivelului de aplicare a redondanței, ca și de durata misiunii (curbele *a*, *b* și *c*). Dacă dimensiunea modulului funcțional — component al sistemului — căruia i se aplică această structură redondantă a variat astfel încît F_n s-a

micșorat de două ori față de valoarea sa maximă, dar atunci cînd F_n crește de 10 ori, indicele α se micșorează de 5 ori. Această observație subliniază că este de dorit ca structura redondantă să fie aplicată la nivel optimal, obținîndu-se în acest fel o creștere importantă a fiabilității prin aplicarea redondanței.

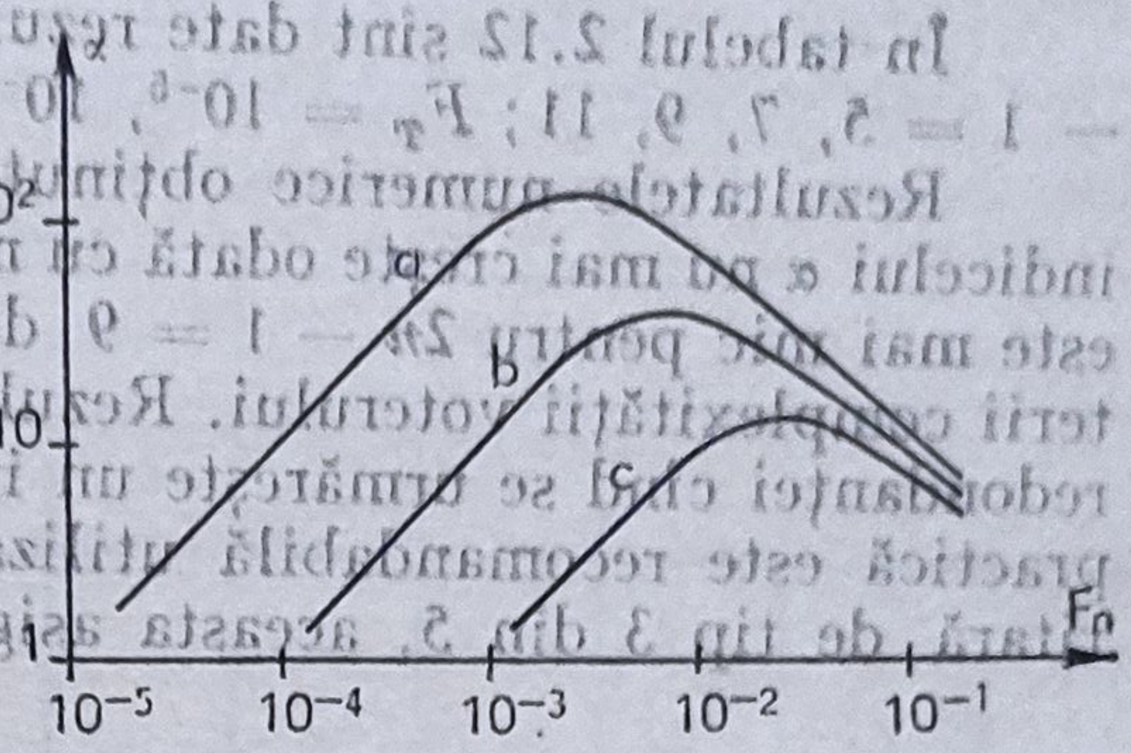


Fig. 2.39. Indicele α de îmbunătățire a fiabilității pentru structura redondantă logică majoritară implementată la nivel neiptimal: $a \rightarrow F_T = 10^{-5}$; $b \rightarrow F_T = 10^{-4}$; $c \rightarrow F_T = 10^{-3}$.

Se remarcă de asemenea că dimensiunea nivelurilor optimale cărora li se implementează redondanța variază în funcție de timpul alocat misiunii; cu cît timpul misiunii este mai mare, cu atît dimensiunile acestor module trebuie să fie mai mari comparativ cu cazul stabilirii nivelului optimal pentru timpi mici ai misiunii, cînd se utilizează componente de același tip.

A. 4. *Analiza structurii redondante logică majoritară simplă de ordinul* n *din* 2n—1

Prin extrapolarea rezultatelor obținute în § A 2, se obțin următoarele ecuații pentru cazul în care ordinul redondanței este n din $2n-1$:

$$F_s = \frac{C_{2n-1}^{n}}{2} F_T^{n} + \frac{C_{2n-1}^{n}}{2} F_n^{n}, \tag{2.61}$$

$$\alpha = \frac{2^{n-1} F_n}{C_{2n-1}^{n} (2^{n-2} F_T + F_n^{n})}; \tag{2.62}$$

$$I_{cp} = 2n - 1 + \frac{n_T \left(C_{2n-1}^{n} + 1\right) F_T}{F_n}, \tag{2.63}$$

$$I_{ci} = 2n - 1 + \frac{C_{2n-1}^{n} (n + 1) F_T}{F_n}. \tag{2.64}$$

Indicele α va fi maxim atunci cînd

$$F_n = 2^{(n-1)/n} \cdot \left(\frac{F_T}{2(n-1)}\right)^{1/n}, \tag{2.65}$$

astfel încît:

$$\alpha_{max} = \frac{2^{(2n-1)/n} [2(n-1)]^{(n-1)/n}}{C_{2n-1}^{n} \cdot 2 n F_T^{(n-1)/n}}; \tag{2.66}$$

$$I_{cp\ opt} = 2n - 1 + n_T (C_{2n-1}^{n} + 1) \left(\frac{F_T}{2}\right)^{(n-1)/n} \cdot (2n - 2)^{1/n}; \tag{2.67}$$

$$I_{ci\ opt} = 2n - 1 + C_{2n-1}^{n} \cdot (n + 1) (2n - 2)^{1/n} \left(\frac{F_T}{2}\right)^{(n-1)/n}. \tag{2.68}$$

În tabelul 2.12 sînt date rezultatele numerice obținute în cazul $2n - 1 = 5, 7, 9, 11$; $F_T = 10^{-5}, 10^{-4}, 10^{-3}$ și $n_T = 3$.

Rezultatele numerice obținute aici reliefează că valoarea maximă a indicelui α nu mai crește odată cu mărirea gradului redondanței și că acesta este mai mic pentru $2n - 1 = 9$ decît pentru $2n - 1 = 5$, din cauza creșterii complexității voterului. Rezultă că nu este necesară creșterea gradului redondanței cînd se urmărește un indice α_{max} de valoare mare; de aceea, în practică este recomandabilă utilizarea structurii redondante logică majoritară de tip 3 din 5, aceasta asigurînd mărirea apreciabilă a fiabilității.

B. Determinarea indicilor de creștere a fiabilității, de cost și eficiență pentru structura redondantă logică majoritară multiplă

Această structură redondantă [39] este o dezvoltare a structurii redondante logică majoritară simplă în sensul introducerii unui număr de votere redondante (fig. 2.8); pentru structura redondantă logică majoritară de tip 2 din 3 se introduc trei votere, ceea ce înseamnă o ieșire triplată. De aceea un subsistem cu această structură impune și celorlalte subsisteme cu care este conectat — chiar dacă au sau nu structură redondantă — intrări triplate. Rezultă că interconexiunile unui sistem redondant de acest tip urmează același aranjament ca la sistemul redondant, cu excepția faptului că ele sînt triplate.

B 1. *Determinarea indicilor de îmbunătățire a fiabilității, de cost și eficiență, utilizîndu-se modelul general de evaluare a fiabilității*

Pentru dezvoltările ulterioare se consideră un modul funcțional cu o singură intrare, căruia i se aplică această structură redondantă, ceea ce implică o intrare triplată pentru sistemul redondant. Analizîndu-se funcționarea schemei, se observă că apariția unui defect la o intrare precum și defectarea unui modul care nu corespunde tripletului vor conduce la defectarea schemei. Un defect la o intrare a tripletului se datorează — în primul rînd — defectării unui voter de la subsistemul anterior celui considerat în structura sistemului.

• Pentru o astfel de structură redondantă, mulțimea tăieturilor minimale este dată în tabelul 2.13.

Tabelul 2.12. PERFORMANȚELE STRUCTURII REDONDANTE LOGICĂ MAJORITARĂ n DIN $2n-1$

$2n-1$	$FT = 10^{-5}$				$FT = 10^{-4}$				$FT = 10^{-3}$			
	5	7	9	11	5	7	9	11	5	7	9	11
α_{max}	287	259	146	64,3	61,9	46,1	23,1	9,43	13,3	8,16	3,66	1,38
$I_{cp\ opt}$	5,02	7,02	9,03	11,1	5,07	7,1	9,21	11,5	5,33	7,57	10,3	14,6
$I_{cp\ opt}$	5,02	7,03	9,07	11,2	5,09	7,16	9,42	12,2	5,4	7,92	11,6	19,4
$F_{n\ opt}$	0,0215	0,0604	0,115	0,252	0,0464	0,107	0,182	0,262	0,10	0,191	0,289	0,384

Tabelul 2.13. MULȚIMEA TĂIETURILOR MINIMALE PENTRU STRUCTURA REDONDANTĂ LOGICĂ MAJORITARĂ MULTIPLĂ DE TIP 2 DIN 3

Numărul de ordine al tăieturii minimale	Elemente defecte									Probabilitate de apariție
	Triplet intrare			Modul funcțional			Votere			
	1	2	3	1	2	3	1	2	3	
1							×	×		
2							×		×	$3\,F_v^2$
3								×	×	
4				×	×					
5				×		×				$3\,F_n^2$
6					×	×				
7	×					×				
8	×						×			
9		×		×						
10		×					×			$6\,F_nF_v$
11				×	×					
12				×		×				

• Funcția de repartiție a timpului de funcționare pentru această structură redondantă, evaluată cu modelul tăieturilor minimale, este:

$$F_r = 3\,F_n^2 + 3\,F_v^2 + 6\,F_v\,F_n. \tag{2.69}$$

• Procedîndu-se în mod analog ca la studiul A, rezultă:

$$\alpha = \frac{F_n}{3\,(F_n^2 + F_v^2 + 2\,F_vF_n)}; \tag{2.70}$$

$$F_{n\,opt} = F_v; \tag{2.71}$$

$$\alpha_{max} = \frac{1}{12\,F_v}. \tag{2.72}$$

• Pe baza aceloraşi considerente folosite în situaţia studiată anterior se obţin următoarele expresii pentru indicele de cost, I_c:

$$I_c = 3 + \frac{3\,F_v}{F_n}, \tag{2,73}$$

respectiv pentru indicele eficienţă, E:

$$E = \frac{F_n^2}{9\,(F_n^2 + F_v^2 + 2\,F_v\,F_n)\,(F_v + F_n)} \tag{2.74}$$

B.2. Determinarea indicilor de îmbunătăţire a stabilităţii, de cost şi eficienţă utilizînd modelul modurilor de defectare

Fie o structură redondantă logică majoritară multiplă de tipul celei indicate în figura 2.8. Schema voterelor V este dată în figura 2.33.

Referitor la modurile de defectare ale sistemului se fac aceleaşi consideraţii ca în § A.2; în plus, se ţine seama de observaţia formulată anterior, cu privire la determinarea mulţimii de defectări minimale necesară pentru evaluarea parametrului F_r.

În cazul modelului *modurilor de defectare* — abordat în continuare — se apreciază că defectarea sistemului cu această structură redondantă este determinată de defectarea în acelaşi mod a două dintre modulele funcţionale ale structurii redondante (blocare în „1" sau în „0"). Probabilitatea acestei mulţimi de defectări posibile este $6(F_n/2)^2$. O altă mulţime de defectări minimale se referă la scurtcircuitarea joncţiunii emitor-bază a tranzistorului multiemitor al porţii NAND finale din oricare circuit al unuia dintre votere; probabilitatea acestei mulţimi de defectări este $3 \cdot 9F_{TS}^2$. În sfîrşit, o ultimă mulţime de defectări — care conduce la defectarea sistemului redondant — este realizată de o intrare defectă a tripletului de intrări, precum şi de prezenţa aceluiaşi tip de defectare într-un alt modul al tripletului, care nu primeşte respectiva intrare. Defectarea pe intrarea primară se datoreşte porţii finale a circuitului voterului de la un alt subsistem redondant al sistemului, conectat la intrarea analizată; probabilitatea acestui eveniment este: $3 \cdot 6 F_{TS} \cdot \frac{F_n}{2}$.

Luîndu-se în consideraţie toate aceste mulţimi de defectări minimale, rezultă următoarea expresie a funcţiei de repartiţie a timpului de funcţionare pentru structura redondantă analizată:

$$F_r = 27 F_{TS}^2 + \frac{3}{2} F_n^2 + 9 F_{TS} F_n. \tag{2.75}$$

În ipoteza $F_{TS} = F_T/2$, ecuaţia (2.75) devine:

$$F_r = 6{,}75 F_T^2 + 4{,}5 F_T F_n + 1{,}5 F_n^2, \tag{2.76}$$

astfel încît:

$$\alpha = \frac{F_n}{6{,}75 F_T^2 + 4{,}5 F_T F_n + 1{,}5 F_n^2}, \tag{2.77}$$

de unde se obţine:

$$F_{n\,opt} = \sqrt{4{,}5}\, F_T, \tag{2,78}$$

iar

$$\alpha_{max} = \frac{1}{6{,}75 (\sqrt{2} + 1) F_T}. \tag{2.79}$$

Raţionîndu-se în cazul studiului A, rezultă următoarea expresie pentru indicii de cost ai acestei structuri redondante:

$$I_{cp} = 3 + 12 n_T F_T / F_n, \tag{2.80}$$

iar

$$I_{ci\,opt} = 3 + 2\sqrt{F_p/F_v}, \tag{2.81}$$

și

$$I_{cr\,opt} = 3 + 4\sqrt{2\,n_m}, \tag{2.82}$$

$$I_{ci\,opt} = 3 + 9\sqrt{2}. \tag{2,83}$$

B.3. *Rezultate numerice*

1°. *Structura redondantă implementată la nivel optimal.* Implementându-se această structură redondantă la nivelul optimal stabilit în conformitate cu criteriul „indicele α este de valoare maximă", se obțin pentru α_{max}, $F_{n\,opt}$, $I_{c\,opt}$, E_{opt} — atunci cînd se consideră $F_v = 9\,F_T$, $n_p = 3$, iar F_r se determină cu modelul uzual de evaluare a fiabilității — valorile numerice cuprinse în tabelul 2.14. Valorile numerice ale parametrilor α_{max}, $I_{cr\,opt}$, $I_{ci\,opt}$, $F_{n\,opt}$, E_{opt}, atunci cînd pentru evaluarea lui F_r se utilizează modelul modurilor de defectare, sînt cuprinse în tabelul 2.15.

Din analiza acestor elemente se observă că dimensiunea unității optime, în termenii numărului de tranzistoare echivalente pe poartă, este independentă de probabilitatea de defectare a tranzistorului echivalent.

Tabelul 2.14. PERFORMANȚELE OPTIMALE ALE STRUCTURII REDONDANTE LOGICĂ MAJORITARĂ MULTIPLĂ DE TIP 2 DIN 3 CALCULATE CU MODELUL UZUAL DE EVALUARE A FIABILITĂȚII

F_T		10-6	10-5	10-4	10-3
	α_{max}	9259,2	925,92	92,592	9,2592
	$I_{c\,opt}$	6	6	6	6
	$F_{n\,opt}$	$9 \cdot 10^{-6}$	$9 \cdot 10^{-5}$	$9 \cdot 10^{-4}$	$9 \cdot 10^{-3}$
	E_{opt}	1543,2	154,32	15,432	1,5432

Tabelul 2.15. PERFORMANȚE OPTIMALE ALE STRUCTURII REDONDANTE LOGICĂ MAJORITARĂ MULTIPLĂ DE TIP 2 DIN 3, CALCULATE CU MODELUL MODURILOR DE DEFECTARE

F_T		10-6	10-5	10-4	10-3
	α_{max}	61 395,8	6 139,5	613,95	61,39
	$I_{cp\,opt}$	19,95	19,95	19,95	19,95
	$I_{ci\,opt}$	15,8	15,8	15,8	15,8
	$F_{n\,opt}$	$2,123 \cdot 10^{-6}$	$2,123 \cdot 10^{-5}$	$2,123 \cdot 10^{-4}$	$2,123 \cdot 10^{-3}$
	$E_{p\,opt}$	23 076,55	2307,65	230,76	23,07
	$E_{i\,opt}$	23 906,55	2390,65	239,06	23,9

Acest rezultat este *contrar* celui obținut la structura redondantă logică majoritară simplă; deosebirea se explică prin faptul că voterul pentru structura redondantă logică majoritară multiplă este multiplicat de același număr de ori ca și sistemul căruia i se aplică redondanța.

2°. *Structura redondantă aplicată la alt nivel decît cel optimal.* În cele ce urmează sînt indicate rezultate numerice reprezentative — bazate pe dezvoltările teoretice anterioare — ale indicilor de îmbunătățire a fiabilității, de cost și eficiență, pentru diferite valori ale parametrilor F_n și F_T; astfel, valorile din tabelul 2.16 sînt obținute utilizîndu-se modelul uzual de evaluare a fiabilității structurii logice majoritare, iar cele din tabelul 2.17 —, folosindu-se modelul modurilor de defectare în aprecierea fiabilității sistemului.

Pe baza rezultatelor numerice obținute se pot face următoarele observații:

— pentru nivelul optimal de implementare a structurii redondante, la care indicele α are valoare maximă, indicele de cost este mult mai mare decît cel obținut pentru oricare alt nivel de implementare.

— se poate obține o reducere a indicelui de cost cu sacrificarea în mică măsură a indicelui α, considerîndu-se o altă partiționare a sistemului în

Tabelul 2.16. PERFORMANȚE ALE STRUCTURII REDONDANTE LOGICĂ MAJORITARĂ MULTIPLĂ DE TIP 2 DIN 3 CALCULATE CU MODELUL UZUAL DE EVALUARE A FIABILITĂȚII

Fn	F_T =10-5					F_T = 10-4				F_T = 10-3		
	9.10-5	2.10-4	10-3	10-2	10-1	9.10-4	2.10-3	10-2	10-1	9.10-3	2.10-2	10-1
α	926	793	282	33	3,3	92,6	79,3	28,2	3,3	9,26	7,93	2,82
I_c	6	4,35	3,3	3,03	3	6	4,35	3,3	3,03	6	4,35	3,3
E	154,3	182,2	86,3	10,8	1,1	15,43	18,2	8,64	1,08	1,54	1,8	0,86

Tabelul 2.17. PERFORMANȚE ALE STRUCTURII REDONDANTE LOGICĂ MAJORITARĂ MULTIPLĂ DE TIP 2 DIN 3 CALCULATE CU MODELUL MODURILOR DE DEFECTARE

Fn	F_T = 10-5					F_T = 10-4				F_T = 10-3		
	2,1.10-5	10-4	10-3	10-2	10-1	2,1.10-4	10-3	10-2	10-1	2,10-3	10-2	10-1
α	6139 .5	4 958	647	66,4	6,6	614	496	64,7	6,7	61,4	49,6	6,5
I_{cp}	19,95	6,6	3,36	3,04	3	19,95	6,6	3,36	3,04	19,95	6,6	3,36
I_{ci}	15,8	5,7	3,27	3,02	3	15,8	5,7	3,27	3,02	15,8	5,7	3,27
E_p	308	751,2	192,6	21,8	2,2	30,8	75	19,3	2,2	3,08	7,5	1,93
E_i	388	869,8	198	22,0	2,2	38,8	87	19,8	2,2	3,9	8,6	1,98

vederea aplicării redondanței. De exemplu, pentru $F_T = 10^{-4}$, $\alpha_{max} = 614$, iar $I_{cp} = 20$, o reducere a indicelui α la 496 (cu aproximativ 18%) conduce la o scădere importantă a valorii indicelui I_{cp}.

Ținîndu-se seama de aceste rezultate, se poate concluziona aprecîndu-se că nivelul optim de implementare a redondanței — determinat conform criteriului „indicele α este de valoare maximă" — nu este cel mai eficient nivel de implementare a redondanței. De aceea, se consideră că un criteriu mai bun în vederea găsirii optimului implementării ar fi cel al lui E_{max}, care, pentru acest tip de structură redondantă, nu mai coincide cu α_{max} [5].

3°. În figura 2.34 este ilustrată variația indicelui α cu F_T, determinată cu modelul uzual de evaluare a fiabilității (curba *b*) și pe baza modelului modurilor de defectare (curba *a*); apare clar că indicele α determinat cu modelul uzual de evaluare a fiabilității este practic cu un ordin de mărime mai mic decît atunci cînd este determinat cu modelul modurilor de defectare.

În figura 2.35 este prezentată variația indicelui α în funcție de F_n pentru $F_T = 10^{-5}$, 10^{-4} și 10^{-3}. Aceste reprezentări grafice permit proiectanților de sistem o alegere rapidă a gradului de partiționare a sistemului în vederea aplicării redondanței pentru a obține o creștere cît mai mare a fiabilității.

C. Determinarea indicilor de îmbunătățire a fiabilității, de cost și eficiență pentru structura redondantă dinamică (de comutație)

Se consideră un sistem cu structură redondantă de comutație de tip activ — unitatea de rezervă este în funcțiune — cu o singură unitate de rezervă (fig. 2.36*a*). Fiecare dintre cele două module ale sistemului primește

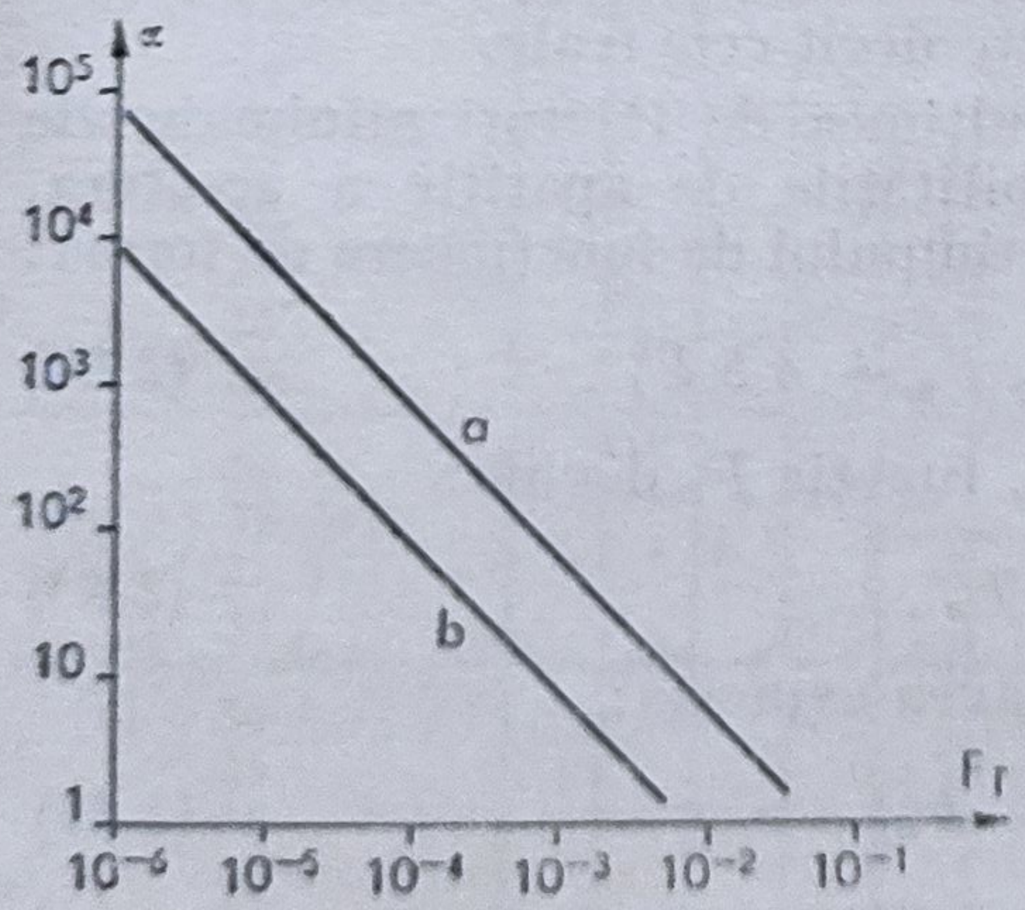


Fig. 2.34. Indicele α de îmbunătățire a fiabilității pentru structura redondantă logică majoritară multiplă:

a — funcția de fiabilitate evaluată cu modelul modurilor de defectare; *b* — funcția de fiabilitate evaluată cu modelul uzual.

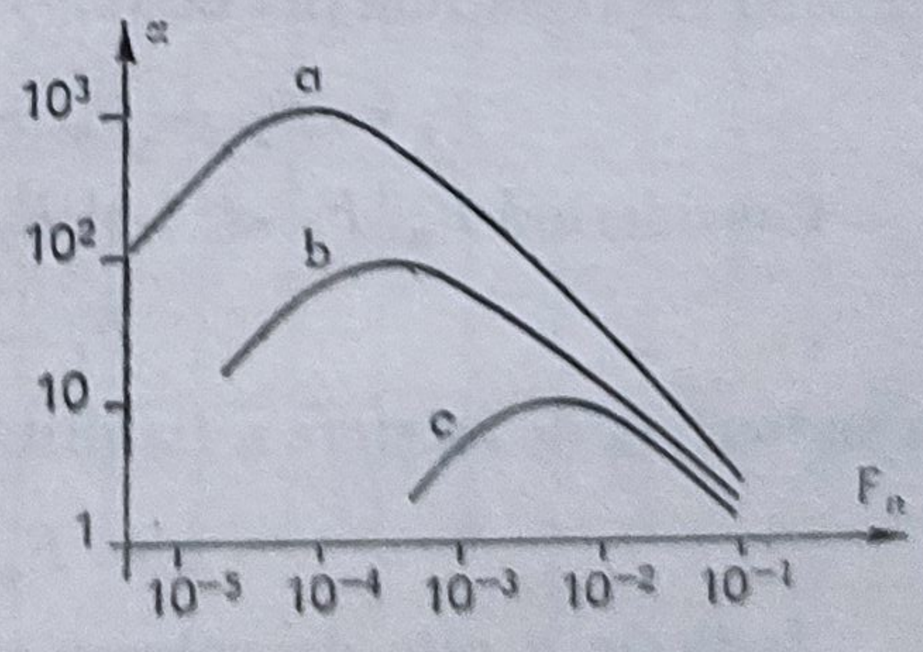


Fig. 2.35. Indicele α de îmbunătățire a fiabilității pentru structura redondantă multiplă implementată la nivel neoptimal:

$a - F_T = 10^{-3}$; $b - F_T = 10^{-4}$; $c - F_T = 10^{-5}$.

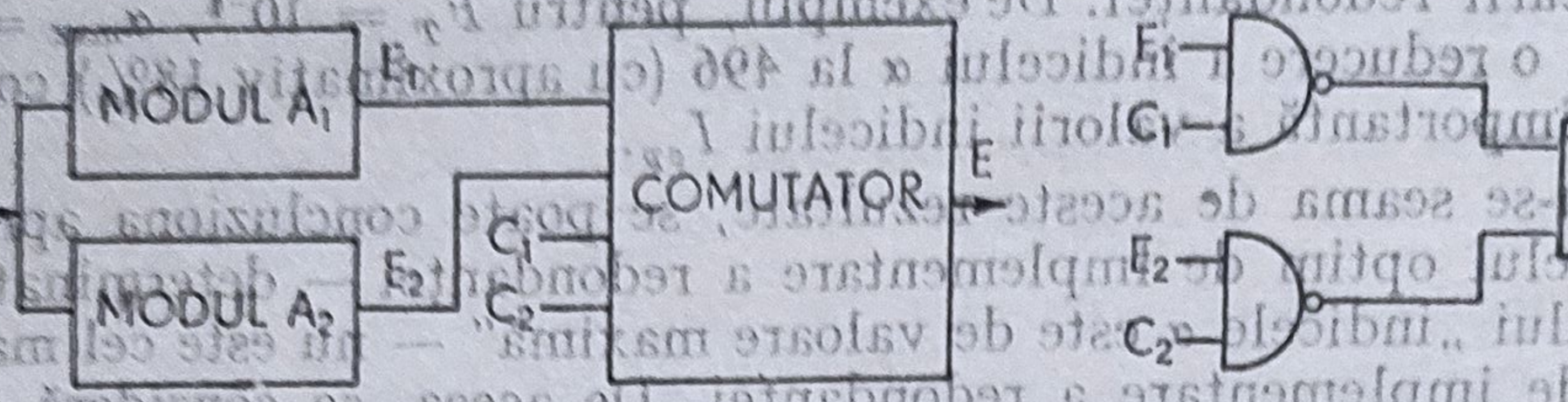


Fig. 2.36. Structură redondantă dinamică (de comutație) de gradul 1:
a — schema structurii; b — schema electrică a circuitului comutator de activare.

aceleași semnale la intrare, iar semnalul-răspuns de ieșire — este selectat dintre semnalele de ieșire ale celor două module funcționale în conformitate cu comanda sistemului de monitorizare, care detectează modulul funcțional defect și comandă comutatorul.

În figura 2.36*b* se propune [29] o schemă de comutator, ce poate fi utilizabilă pentru acest tip de redondanță. Subsistemul de monitorizare acționează comenzile C_1 sau C_2 în funcție de modulul în care a fost localizat defectul, semnalul de ieșire al sistemului fiind astfel corectat.

Un subsistem cu o asemenea structură redondantă se va conecta în sistem exact în același aranjament ca la sistemul neredondant, pentru că, așa cum s-a arătat în § 2.2.6, interconexiunile între subsisteme nu sînt redondante.

C. Determinarea indicilor de creștere a fiabilității, de cost și de eficiență, utilizînd-se modelul modulelor defectate

Se consideră că detecția defectării și comanda comutării sînt perfect fiabile; desigur, aceste ipoteze sînt simplificatoare, iar rezultatele obținute în condițiile respective vor fi mai optimiste decît cele reale.

• În tabelul 2.18 sînt indicate mulțimea de tăieturi minimale ale sistemului considerat, precum și probabilitățile de apariție a acestora, rezultînd expresia funcției de repartiție a timpului de funcționare de forma:

$$F_r = F_T + F_n^2 + 4{,}5\, F_T F_n + 4{,}5\, F_T^2 \,. \tag{2.84}$$

Considerînd $F_n, F_T \ll 1$ și $F_T < F_n$, funcția F_r devine:

$$F_r = F_T + F_n^2 \,, \tag{2.85}$$

iar indicele α de creștere a fiabilității va avea expresia:

$$\alpha = F_n / (F_T + F_n^2) \,. \tag{2.86}$$

Indicele α este de valoare maximă pentru

$$F_{n\,\mathrm{opt}} = \sqrt{F_T}, \tag{2.87}$$

iar

$$\alpha_{\max} = 1/2 \sqrt{F_T} \,. \tag{2.88}$$

Tabelul 2.18. MULȚIMEA TĂIETURILOR MINIMALE PENTRU STRUCTURA REDONDANTĂ DINAMICĂ (DE COMUTAȚIE)

Numărul de ordine al tăieturii minimale	Modul funcțional/poarta	M_1		M_2		G_1				G_2				J_1		J_2		Probabilitate de apariție
	Joncțiune					J_1		J_2		J_1		J_2						
	Mod de defectare	1	0	1	0	G	S	G	S	G	S	G	S	G	S	G	S	
1		×			×													$4\left(\frac{F_n}{2}\right)^2$
2		×			×													
3			×	×														
4			×		×													
5															×			$2\,(F_T/2)$
6																×		
7		×		×														$18\frac{F_n}{2}\cdot\frac{F_T}{2}$
8		×			×													
9			×	×														
10			×		×													
11				×	×													
12		×		×														
13			×	×										×				
14			×		×													
15			×	×			×											
16			×		×		×											
17			×	×											×			
18			×		×									×				
19			×	×														
20			×		×													
21		×		×										×				$18\left(\frac{F_T}{2}\right)^2$
22		×			×													
23		×		×			×											
24		×			×		×											
25		×		×			×											
26		×			×										×			
27		×		×										×				
28		×			×													
29		×			×		×											
30		×		×										×				
31		×			×													
32		×		×														
33		×			×													
34		×		×										×				
35		×			×													
36		×		×														
37		×			×													
38		×			×		×											
39		×		×										×				
40		×			×													
41		×		×														
42		×			×													

• Indicele de cost, I_{cp}, are expresia:

$$I_{cp} = 2 + 3\, n_T\, F_T/F_n, \tag{2.89}$$

relație stabilită avîndu-se în vedere faptul că pentru această structură redondantă modulele funcționale sînt dublate, comutatorul conține trei porți, iar modulul neredondant conține $F_n/n_T\, F_T$ porți echivalente.

• Similar, se poate determina expresia indicelui de cost, I_{ci}:

$$I_{ci} = 2 + \frac{6\, F_T}{F_n}. \tag{2.90}$$

• Atunci cînd structura redondantă se implementează la nivel optimal, acești indici devin:

$$I_{cp\ \mathrm{opt}} = 2 + 3\, n_T \sqrt{F_T}\ ; \tag{2.91}$$

$$I_{ci\ \mathrm{opt}} = 2 + 6\, \sqrt{F_T}\ . \tag{2.92}$$

• În continuare, rezultatele date de ecuațiile (2.85) ÷ (2.92) sînt generalizate pentru cazul în care gradul redondanței este $n - 1$ (sînt n module funcționale identice, din care $n - 1$ rezerve). Comutatorul se consideră cu structură identică (fig. 2.36*b*), cu excepția numărului de porți la intrare, egal cu n. Analizîndu-se mulțimea de defectări minimale posibile, rezultă:

$$F_r = F_n^n + 0{,}5\, n\, F_T, \tag{2.93}$$

de unde se obțin:

$$\alpha = \frac{F_n}{F_n^n + 0{,}5\, n\, F_T}; \tag{2.94}$$

$$I_{cp} = n + \frac{(n+1)\, n_T\, F_T}{F_n}\ ; \tag{2.95}$$

$$I_{ci} = n + \frac{3\, n\, F_T}{F_n}; \tag{2.96}$$

$$F_{n\ \mathrm{opt}} = \left[\frac{n\, F_T}{2\,(n-1)}\right]^{1/n}; \tag{2.97}$$

$$\alpha_{\max} = \frac{1}{n}\left[\frac{2\,(n-1)}{n\, F_T}\right]^{(n-1)/n} \tag{2.98}$$

$$I_{cp} = n + (n+1)\, n_T \left[\frac{2\,(n-1)}{n}\right]^{1/n} F_T^{\ (n-1)/n}; \tag{2.99}$$

$$I_{ci\ \mathrm{opt}} = n + 3\, n \left[\frac{2\,(n-1)}{n}\right]^{1/n} F_T^{\ (n-1)/n}. \tag{2.100}$$

C 2. *Rezultate numerice*

1°. *Structura redondantă aplicată la nivel optimal.* Cu ecuațiile (2.87), (2.88), (2.90) și (2.92) se obțin pentru $F_{n\ \mathrm{opt}}$, $\alpha_{\max}$, I_{cp}, I_{ci} rezultatele numerice prezentate în tabelul 2.19, pentru cîteva valori ale lui F_T.

Tabelul 2.19. PERFORMANȚE OPTIMALE ALE STRUCTURII REDONDANTE DINAMICĂ (DE COMUTAȚIE) CU GRADUL REDONDANȚEI $r = 1$

F_T	10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}
$F_{n\ opt}$	10^{-6}	3.10^{-5}	10^{-4}	$3.1.10^{-3}$	10^{-1}
α_{max}	500	158	50	15,8	5
$I_{cp\ opt}$	2,01	2,03	2,09	2,28	2,9
$I_{ci\ opt}$	2,01	2,02	2,06	2,2	2,6
$Ep\ opt$	249	77,83	23,92	6,92	1,72
$Ei\ opt$	249	78,21	24,47	7,81	1,92

Analizîndu-se aceste date, se observă că dimensiunile modulului optim căruia i se aplică această structură redondantă variază de la 10 tranzistoare echivalente, atunci cînd $F_T = 10^{-2}$, la 1 000 tranzistoare echivalente, cînd $F_T = 10^{-6}$.

Cu relațiile (2.97) ÷ (2.100), pentru $n = 3$ și, respectiv, gradul redondanței este doi sau trei; pentru $F_{n\ opt}$, α_{max}, $I_{c\ opt}$ valorile numerice obținute sînt cuprinse în tabelul 2.20; din tabel rezultă că indicele α_{max} crește odată cu gradul redondanței.

Tabelul 2.20. PERFORMANȚE OPTIMALE ALE STRUCTURII REDONDANTE DINAMICE (DE COMUTAȚIE) CU $r > 1$

F_T	$r = 2$				$r = 3$			
	10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-6}	10^{-5}	10^{-4}	10^{-3}
$F_{n\ opt}$	$9,1 \cdot 10^{-3}$	$1,96 \cdot 10^{-2}$	$4,2 \cdot 10^{-2}$	$9 \cdot 10^{-2}$	$2,8 \cdot 10^{-2}$	$5,8 \cdot 10^{-2}$	$9 \cdot 10^{-2}$	$16 \cdot 10^{-2}$
α_{max}	4 033	870	187	40,4	10 590	1 910	339	60,3
$I_{cp\ opt}$	3	3,01	3,03	3,13	4	4	4,02	4,09
$I_{ci\ opt}$	3	3	3,02	3,1	4	4	4,01	4,07
$E_{p\ opt}$	1 344	289	61,71	12,9	2 647,5	477,5	84,32	14,8
$E_{i\ opt}$	1 344	290	61,9	13,03	2 647,5	477,5	84,5	14,8

2°. *Structura redondantă implementată la un nivel diferit de cel optimal.* În tabelul 2.21 sînt prezentate cîteva valori numerice ale parametrilor α, I_c și E pentru $n_T = 3$ și gradul redondanței $r = 1$, evidențiindu-se creșterea fiabilității sistemului cu această structură redondantă pentru diferite niveluri de implementare a redondanței. Se remarcă faptul că maximul indicelui α coincide cu maximul indicelui de eficiență.

3°. În figura 2.37 este reprezentată variația indicelui α_{max} cu F_T, atunci cînd nivelul de implementare a redondanței este cel optimal. Creșterea indicelui α odată cu mărirea gradului redondanței nu este spectaculoasă;

Tabelul 2.20. PERFORMANȚE OPTIMALE ALE STRUCTURII REDONDANTE DINAMICE (DE COMUTAȚIE) CU $r > 1$

F_n	$F_T = 10^{-5}$					$F_T = 10^{-4}$					$F_T = 10^{-3}$		
	10^{-4}	10^{-3}	10.8^{-3}	10^{-2}	10^{-1}	10^{-3}	3.10^{-3}	10^{-2}	3.10^{-2}	10^{-1}	10^{-2}	10^{-2}	10^{-1}
α	9,99	90,9	158	90,9	9,99	9,9	27,5	50	30	9,9	9,09	15,8	9,09
I_{cp}	2,9	2,09	2,03	2,01	2	2,9	2,3	2,09	2,03	2,01	2,9	2,3	2,09
I_{ci}	2,6	2,06	2,02	2,01	2	2,6	2,2	2,06	2,02	2,01	2,6	2,2	2,06
E_p	3,44	43,49	77,83	45,22	4,995	3,4	11.95	23.92	14.77	4,92	3,13	6,86	4,34
E_i	3,84	44,12	78,21	45,22	4,995	3,8	12,5	24.27	14,8	4,92	3,5	7,18	4,41

acest fapt se datorește comutatorului, care nu are o structură redondantă și a cărui complexitate se mărește odată cu creșterea gradului redondanței sistemului (tabelul 2.20).

Din reprezentarea grafică a variației indicelui α cu F_n, indicată în figura 2.38, rezultă că atunci cînd F_n variază de zece ori față de valoarea optimului, indicele α se micșorează foarte mult, ceea ce subliniază cerința implementării acestei structuri redondante la nivel optimal, cînd se aduce un cîștig important în ceea ce privește fiabilitatea. Mai mult, analizîndu-se graficul din figura 2.39, care indică variația indicelui α cu I_{cp}, se observă

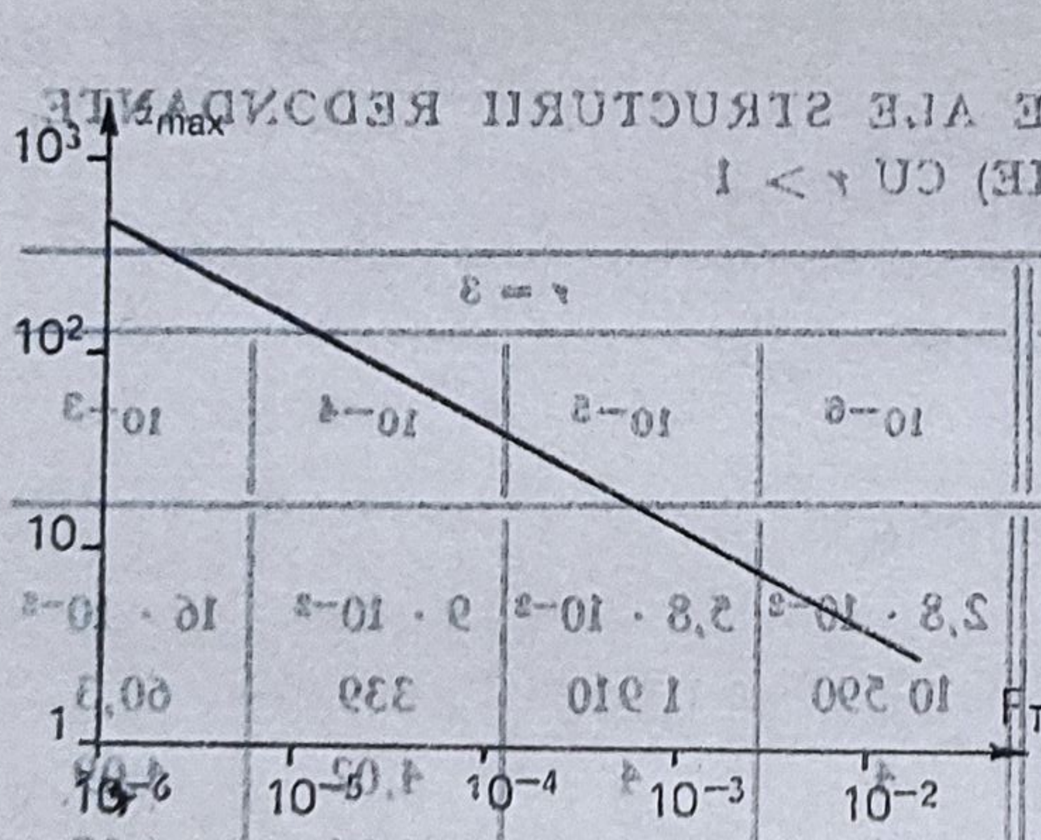


Fig. 2.37. Indicele α_{max} de îmbunătățire a fiabilității pentru structura redondantă dinamică (de comutație).

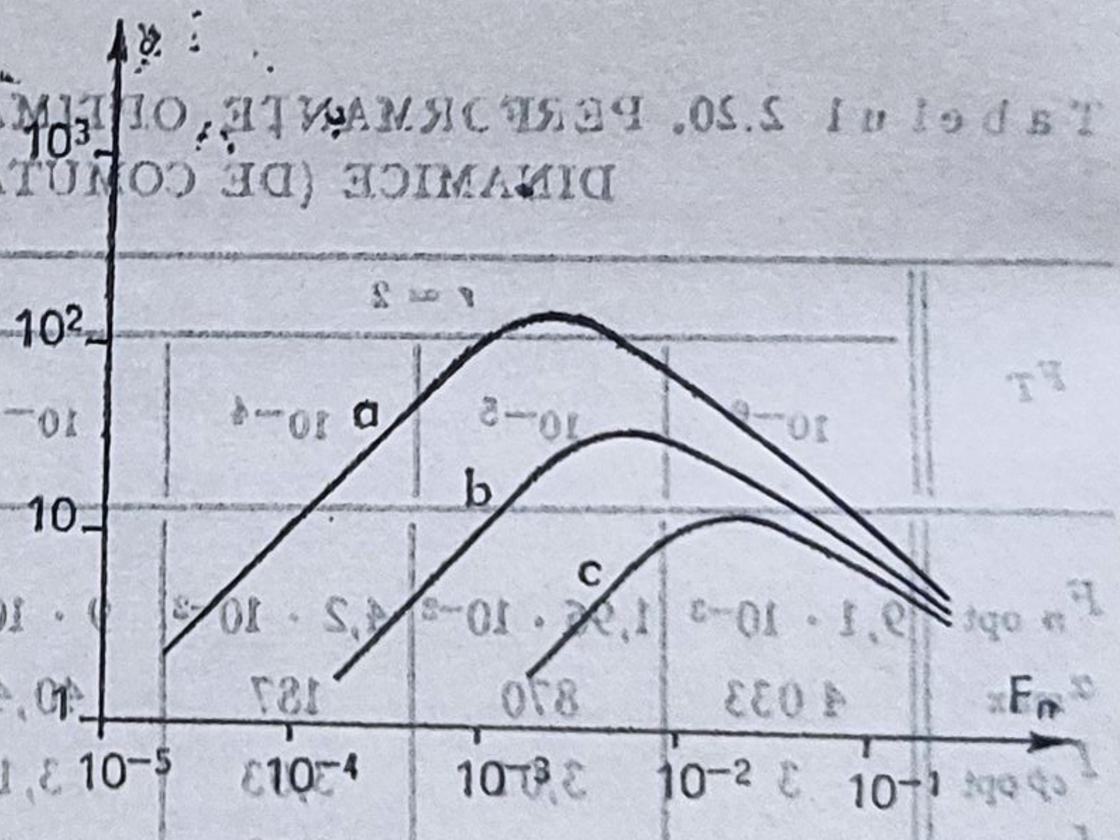


Fig. 2.38. Indicele α de îmbunătățire a fiabilității pentru structura redondantă dinamică (de comutație) implementată la nivel neoptimal:
$a - F_t = 10^{-5}$; $b - FT = 10^{-4}$; $c - FT = 10^{-3}$

Fig. 2.39. Corelația cost-creșterea fiabilității pentru structura redondantă dinamică (de comutație).

cu claritate că pentru această structură redondantă nivelul optimal de dimensionare a partiționării sistemului în vederea aplicării redondanței este cel corespunzător lui α de valoare maximă, la care corespunde un anumit indice I_{cr}; o reducere a indicelui I_{cr}, prin modificarea nivelului partiționării sistemului, nu are sens, întrucît conduce la o scădere accentuată a valorii indicelui α.

2.5.4. CONCLUZII

În acest paragraf, au fost calculați — în vederea evaluării performanțelor structurilor redondante de tip logică majoritară simplă, logică majoritară multiplă și de comutație — indicele logaritmic de creștere a fiabilității, indicii de cost și indicele global de eficiență în conformitate cu dezvoltările propuse în § 2.5.1 și 2.5.2. O analiză comparativă a rezultatelor obținute evidențiază următoarele:

- Pentru structura redondantă logică majoritară simplă de ordinul 2 din 3 — cea mai utilizată pentru implementarea toleranței la defectări în sistemele digitale — *maximul indicelui* α (tabelele 2.9, 2.15 și 2.19) *are o valoare numerică mai mică* comparativ cu celelalte două structuri redondante cu grad de redondanță echivalent. Însă, pentru structura redondantă logică majoritară multiplă de ordinul 2 din 3 *maximului indicelui* α îi corespunde un *indice de cost mult mai mare* (tabelul 2.15) decît cel corespunzător structurii redondante logică majoritară simplă, astfel încît se poate aprecia că structura redondantă logică majoritară multiplă nu este întotdeauna avantajoasă din cauza costului ridicat.

La aceeași valoare a indicelui de cost, valoarea indicelui α — fără a reprezenta maximul lui α — este mai mică pentru structura redondantă logică majoritară multiplă, comparativ cu valoarea indicelui α corespunzător structurii redondante logică majoritară simplă (tabelele 2.11 și 2.17). Dacă factorul de cost este acela care primează, atunci structura redondantă logică majoritară simplă de ordinul 2 din 3 apare ca fiind aceea care trebuie preferată, căci se obține totodată — în anumite condiții — și o creștere importantă a fiabilității.

- Structurii redondante logică majoritară multiplă îi corespund comparativ cu alte structuri redondante, pentru același grad al redondanței, indici de creștere a fiabilității foarte mari, dar totodată și indici de cost mari.

Această structură are performanțe remarcabile pentru valori mari ale funcției de fiabilitate a componentelor sistemului, sau în alți termeni, pentru valori mici ale timpilor alocați misiunii sistemului respectiv. Aceste performanțe descresc marcant odată cu descreșterea valorii funcției de fiabilitate pentru componentele constituente ale sistemului/creșterea timpului alocat misiunii sistemului (tabelul 2.17 și figura 2.35). Rezultatele numerice obținute, subliniază totodată că această tehnică de redondanță aplicată unui sistem mare, fără ca acesta să fie partiționat în module funcționale mai mici, conduce la performanțe de fiabilitate inferioare redondanței de comutație. Un avantaj de ordin constructiv al structurii analizate este acela al aplicabilității sale pentru orice sistem cu intrări și ieșiri binare și faptul că nu este restrictivă numai pentru circuite logice combinaționale și secvențiale

construite cu porți logice, cum ar fi, de exemplu, cazul structurilor redondante logică cuadruplă sau logică prin cablare.

Pentru o structură redondantă logică majoritară multiplă, toate conexiunile între modulele funcționale redondante sînt multiplicate de un număr de ori egal cu cel al modulelor funcționale; din motivul menționat, se cere ca sistemul căruia i se aplică acest tip de redondanță să fie partiționat în module funcționale, care au — preferabil — o singură ieșire, deoarece altfel ar apare un număr mare de conexiuni în sistem; în același timp trebuie ca dimensiunea modulului funcțional să fie cea optimă, pentru a se obține performanțe de fiabilitate maxime și un cost nu prea ridicat. Problema partiționării este facilitată datorită faptului că alegerea dimensiunilor modulului nu afectează critic performanțele schemei redondante (fig. 2.35).

• Structura redondantă dinamică (de comutație) nu trebuie inclusă — de fapt —, în vederea unei analize comparative, în aceeași categorie cu celelalte două structuri redondante, de tip static, deoarece defectul nu este mascat instantaneu la ieșirea sistemului. Ea este însă, o structură redondantă uzuală în construcțiile tolerante la defectări, la fel ca și structura redondantă logică majoritară simplă de tip 2 din 3, motiv pentru care a fost inclusă în prezentul studiu.

Analiza realizată în § 2.5.3. subliniază că performanțele structurii redondante de comutație apar foarte bune la indici de cost mici, dar sînt mediocre la indici de cost mari, comparativ cu celelalte două structuri redondante studiate (tabelul 2.20). O altă caracteristică a acestei structuri redondante, care trebuie avută în atenție pentru implementarea redondanței, este aceea că dimensiunea modulului funcțional ce intră în structura redondantă are un efect critic asupra performanțelor schemei (tabelul 2.21 și figura 2.39), deoarece atît indicele α, cît și indicii de cost sînt puternic influențați de aceste dimensiuni.

• În § 2.5.2 a fost propusă introducerea indicatorului numeric eficiență — un indicator global de analiză a unei structuri redondante, care permite o optimizare globală a implementării tehnicii redondante respective, luîndu-se în considerație atît rezultatele utile ale creșterii fiabilității, dar și efectele economice/financiare ale acestei implementări. Acest indicator a fost calculat în § 2.5.3 pentru cele trei structuri redondante; a rezultat că pentru structura redondantă logică majoritară multiplă valorile maxime ale indicelui eficiență, E, nu corespund valorilor maxime ale indicelui logaritmic de creșterea a fiabilității, α, ca la celelalte structuri redondante, ci la valori mai mici ale acestuia (rezultatele numerice cuprinse în tabelul 2.17). În schimb, pentru structura redondantă de comutație indicatorul eficiență este maxim cînd α este maxim (fig. 2.41), deoarece la o variație mică a lui I_{ep}, indicele α se micșorează foarte mult.

Un avantaj de ordin constructiv al acestor structuri este acela că ele sînt aplicabile oricărui sistem hardware cu intrări și ieșiri binare și nu sînt restrictive pentru circuite cu porți logice.

În analiza întreprinsă s-a urmărit evidențierea unor particularități ale structurilor redondante logică majoritară și de comutație, determinîndu-se — în fiecare caz — un nivel optim de implementare a structurii re-

dondante respective, astfel încît să se poată obține indici de creștere a fiabilității sau de eficiență de valoare maximă. De aceea se poate considera că analiza este utilă proiectanților de sisteme de înaltă fiabilitate.

2.6. IMPLEMENTAREA UNEI STRUCTURI REDONDANTE LA NIVEL OPTIMAL

În paragraful anterior, în urma studiului efectuat în vederea evaluării performanțelor unor structuri redondante, s-a precizat că performanțele acestora sînt dependente atît de nivelul implementării redondanței în sistem, cît și de gradul redondanței.

O serie de cercetări cu privire la variația costului unui sistem în funcție de alocarea fiabilității [13], [43] au evidențiat o descreștere logaritmică a probabilității de defectare a sistemului cu mărirea costului sistemului datorită aplicării redondanței. De aceea o problemă importantă care trebuie soluționată în cazul proiectării sistemelor cu structură redondantă este determinarea gradului optim al redondanței, precum și precizarea nivelului la care trebuie aplicată redondanța, astfel încît fiabilitatea sistemului în prezența diferitelor restricții să fie maximizată; aceste restricții pot fi de natură economică, tehnică (greutate, volum etc.) — condiții importante în special pentru aparatura militară portabilă, aparatura instalată la bordul avioanelor sau rachetelor, pentru echipamentele de telecomunicații etc. — sau pot avea în vedere mentenanța sistemului etc. Problema a fost abordată, în parte, în paragraful anterior.

În [15] au fost analizați și exemplificați mai mulți algoritmi care permit optimizarea structurilor redondante în prezența unor restricții liniare sau neliniare, algoritmi bazați pe metode euristice, directe, cît și pe metoda multiplicatorilor lui Lagrange și programării dinamice.

În cele ce urmează se propune o metodă de analiză a unei strategii de implementare a redondanței, care permite — pe baza caracteristicilor cost — fiabilitate ale sistemului în cauză — să se aleagă o structură redondantă optimală (§ 2.6.1), după care (§ 2.6.2) se analizează nivelul optimal de partiționare a unui sistem în vederea implementării redondanței.

2.6.1. STRATEGII DE IMPLEMENTARE A REDONDANȚEI

Se ridică următoarea problemă importantă a implementării redondanței unui sistem: pentru a se obține o creștere semnificativă a funcției de fiabilitate, se mărește gradul redondanței — prin alocarea unei rezerve suplimentare sistemului redondant respectiv — sau se partiționează în alt mod sistemul, aplicîndu-i-se același tip de redondanță și de același grad, dar la un alt nivel.

Următoarele două exemple vor evidenția importanța unei analize apriori a strategiilor de implementare a redondanței, astfel încît să poată fi adoptată strategia cea mai eficientă pentru condițiile date.

Exemplul următor arată că sistemul partiționat în două module cu valorile funcțiilor de fiabilitate în relația $R_1(t) \leq R_2(t)$ poate avea două structuri redondante, (*a*) și (*b*) (fig. 2.40). Intuitiv, se poate afirma că sistemul (*b*) este mai fiabil decît sistemul (*a*), ceea ce este ușor de verificat. Într-adevăr,

$$R_{(b)}(t) - R_{(a)}(t) = 2R_1(t)R_2(t) - R_1(t)^2 R_2(t)^2 - 2R_1(t)R_2(t) + R_1(t)R_2(t)^2 = R_1(t)R_2(t)^2(1 - R_1(t)) > 0, \forall R_1(t), R_2(t) \in [0, 1]. \quad (2.101)$$

O concluzie identică se obține dacă gradul redondanței crește la 2 (fig. 2.41). Într-adevăr, $R_b - R_a > 0$ atunci cînd

$$3R_1(t) - 3R_2(t) + R_1(t)^2 R_2(t) > 0, \quad (2.102)$$

ceea ce este adevărat, căci întotdeauna

$$R_2(t) < 3/(R_1(t) + 1), \forall R_1(t) \in [0; 1].$$

Deci, în asemenea situație o alocare a unei rezerve pentru subsistemul 1 va conduce la creșterea fiabilității întregului sistem.

Exemplul 2. Se consideră structurile redondante prezentate în figurile 2.40*b* și 2.41*a*, aplicate aceluiași sistem. Nu se poate decide apriori asupra valorii funcției de fiabilitate corespunzătoare celor două structuri. Intuitiv, s-ar putea crede că sistemul din figura 2.41*a* va fi mai fiabil decît sistemul prezentat în figura 2.40*b*. În continuare se va arăta că aceasta nu este întotdeauna adevărat. Sistemul din figura 2.40*b* se notează cu sistemul

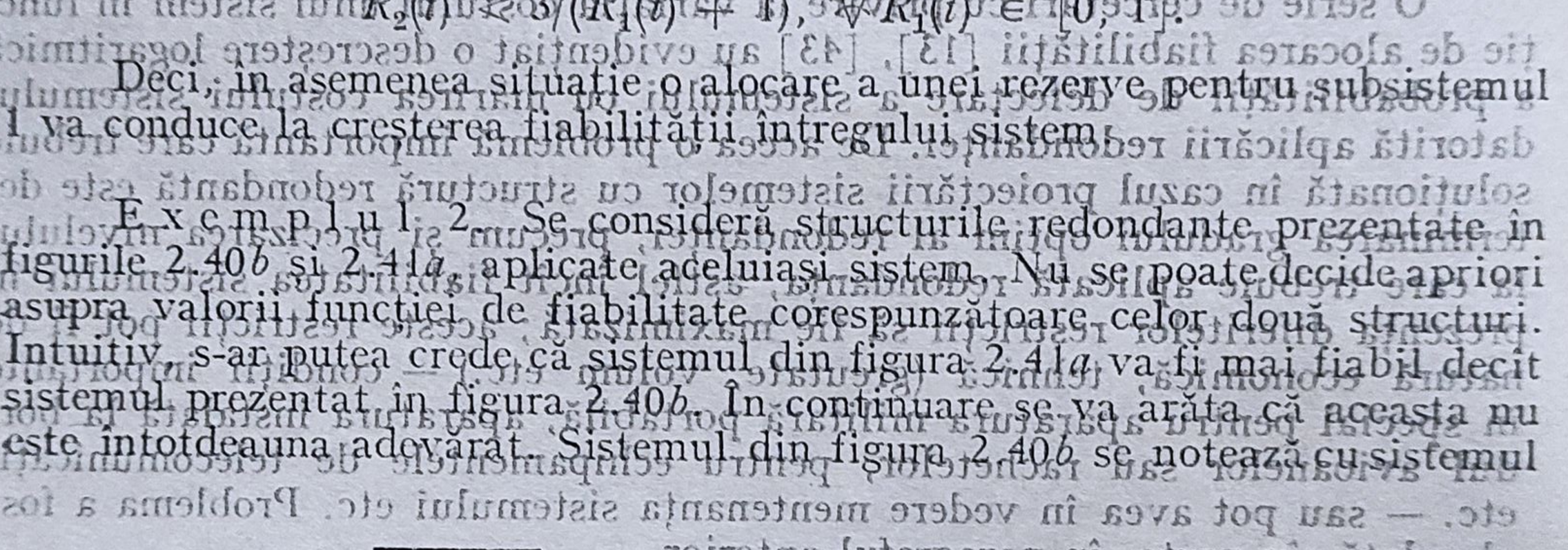


Fig. 2.40. Structuri redondante dinamice (de comutație) implementate unui sistem:
a — structură redondantă se aplică numai elementului 2; *b* — structură redondantă se aplică întregului sistem.

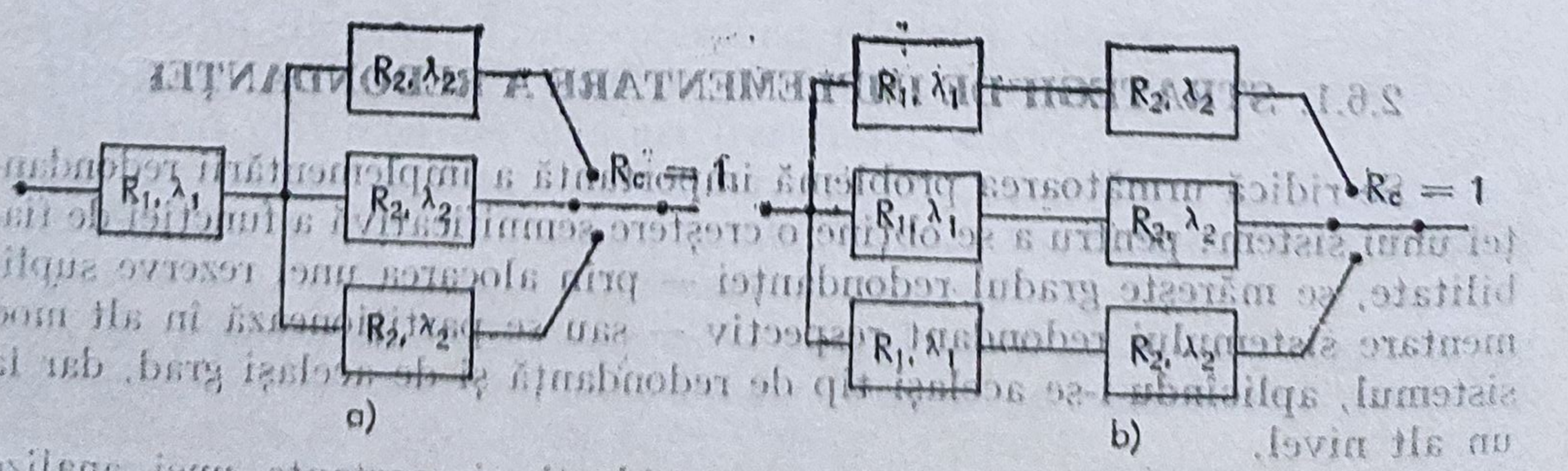


Fig. 2.41. Structuri redondante dinamice (de comutație) de gradul 2 implementate unui sistem:
a — structura redondantă se aplică numai elementului 2; *b* — structură redondantă se aplică întregului sistem.

(I), iar cel din figura 2.41*a* cu sistemul (II). Sînt situații cînd sistemul (II) este mai puțin fiabil decît (I), cu toate că acesta este mai scump. Astfel,

$$R_{II}(t) - R_I(t) = R_1(t)\,(R_2(t)^3 - 3R_2(t)^2 + 3R_2(t) - 2R_2(t) + \\ + R_1(t)\,R_2(t)^2) = R_1(t)R_2(t)\,(R_2(t)^2 - 3R_2(t) + R_1(t)R_2(t) + 1). \quad (2.103)$$

Semnul acestei diferențe se poate studia grafic, analizîndu-se reprezentarea funcției:

$$R_1(t) = (3R_2(t) - R_2^2(t) - 1)/R_2(t) = f(R_2(t)) \quad (2.104)$$

în planul $R_1(t)$, $R_2(t)$, intersectat cu zona $R_1(t) > R_2(t)$ din acest plan (fig. 2.42). Se găsesc condițiile pentru care este îndeplinită inegalitatea $R_{II}(t) > R_I(t)$ sau $R_I(t) > R_{II}(t)$.

Zona notată $\ominus$ (fig. 2.42) indică situațiile în care sistemul (II), cu toate că este mai scump decît (I), este mai puțin fiabil decît acesta, iar zona notată cu $\oplus$ — în care sistemul (II) este mai fiabil decît sistemul (I).

Dreapta $R_1(t) = 1$ în planul (R_1, R_2) este tangentă în punctul de coordonate (1, 1) la curba $R_1 = f(R_2)$. Apare clar că pentru valori apropiate de 1 ale funcțiilor de fiabilitate R_1 și R_2 schema (II) nu este întotdeauna mai fiabilă decît schema (I), ceea ce înseamnă că o structură redondantă de gradul 1, deci mai ieftină, poate fi în anumite situații mai fiabilă decît o structură redondantă de gradul 2.

În continuare se va dezvolta o metodă care permite stabilirea unei strategii optimale de implementare a redondanței.

Problema creșterii fiabilității sistemului al cărui model structural de fiabilitate este reprezentat în figura 2.43*a* poate fi rezolvată în două moduri: fie adoptînd o altă partiționare a sistemului și aplicîndu-i-se acestuia aceeași structură redondantă — strategia I (fig. 2.43*b*) — fie prin creșterea gradului redondanței (alocarea unui element de rezervă suplimentar) — strategia II (fig. 2.43*c*).

Alegerea uneia dintre cele două strategii se poate face prin compararea raportului dintre variația funcției de fiabilitate și variația costului.

În cele ce urmează se presupune că sistemul S are timpul de funcționare repartizat după o lege exponențială. În cazul strategiei I de implementare a redondanței (fig. 2.43*b*) se consideră sistemul partiționat în două subsisteme, S_1 și S_2, cu ratele de defectare λ_1 și λ_2, astfel încît

$$\lambda_1 + \lambda_2 = \lambda_n, \quad (2.105)$$

unde λ_n este rata de defectare a sistemului neredondant. Mărimea subsistemelor S_1 și S_2 poate fi astfel aleasă încît λ_1 să varieze de la λ_n la 0, în timp ce λ_2 variază între 0 și λ_n.

Cu aceste date se poate scrie expresia funcției de fiabilitate a sistemului cu structură redondantă din figura 2.43*b*:

$$R_{SR}(t) = R_1(t)\,(2R_2(t) - R_2(t)^2) = R_1(t)\;R_2(t)\,(2 - R_2(t)) = \\ = e^{-\lambda_n t}\,[2 - e^{(-\lambda_n + \lambda_1)t}]. \quad (2.106)$$

Costul C alocat sistemului cu structură redondantă considerat (fig. 2.43*b*) depinde de timpul misiunii, T, și de ratele de defectare, λ_1 și λ_2, după legea [11]:

$$C = k\,(\lambda_1 + 2\lambda_2)\,T, \quad (2.107)$$

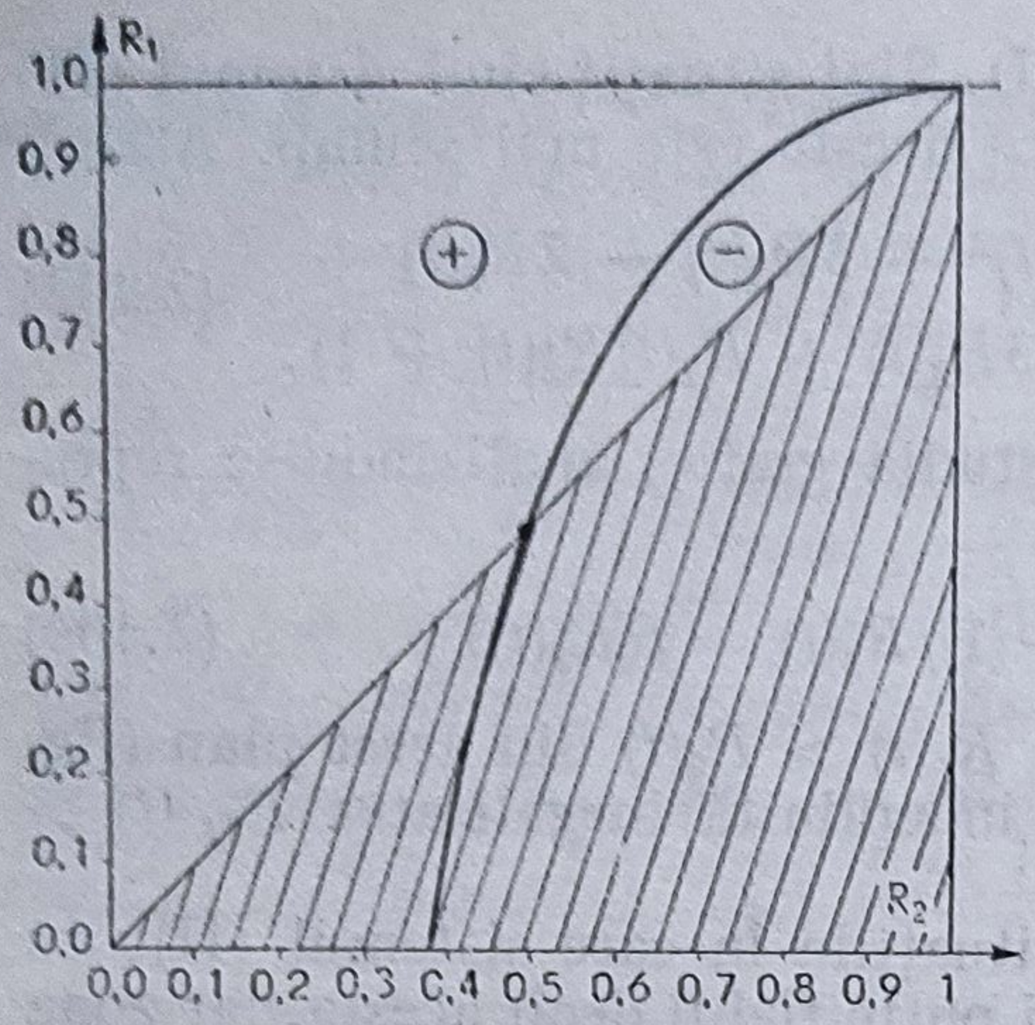


Fig. 2.42. Graficul funcției $R_1 = (3R_2 - R_2^2 - 1)/R_0$ în analiza creșterii fiabilității pentru structurile (I) și (II).

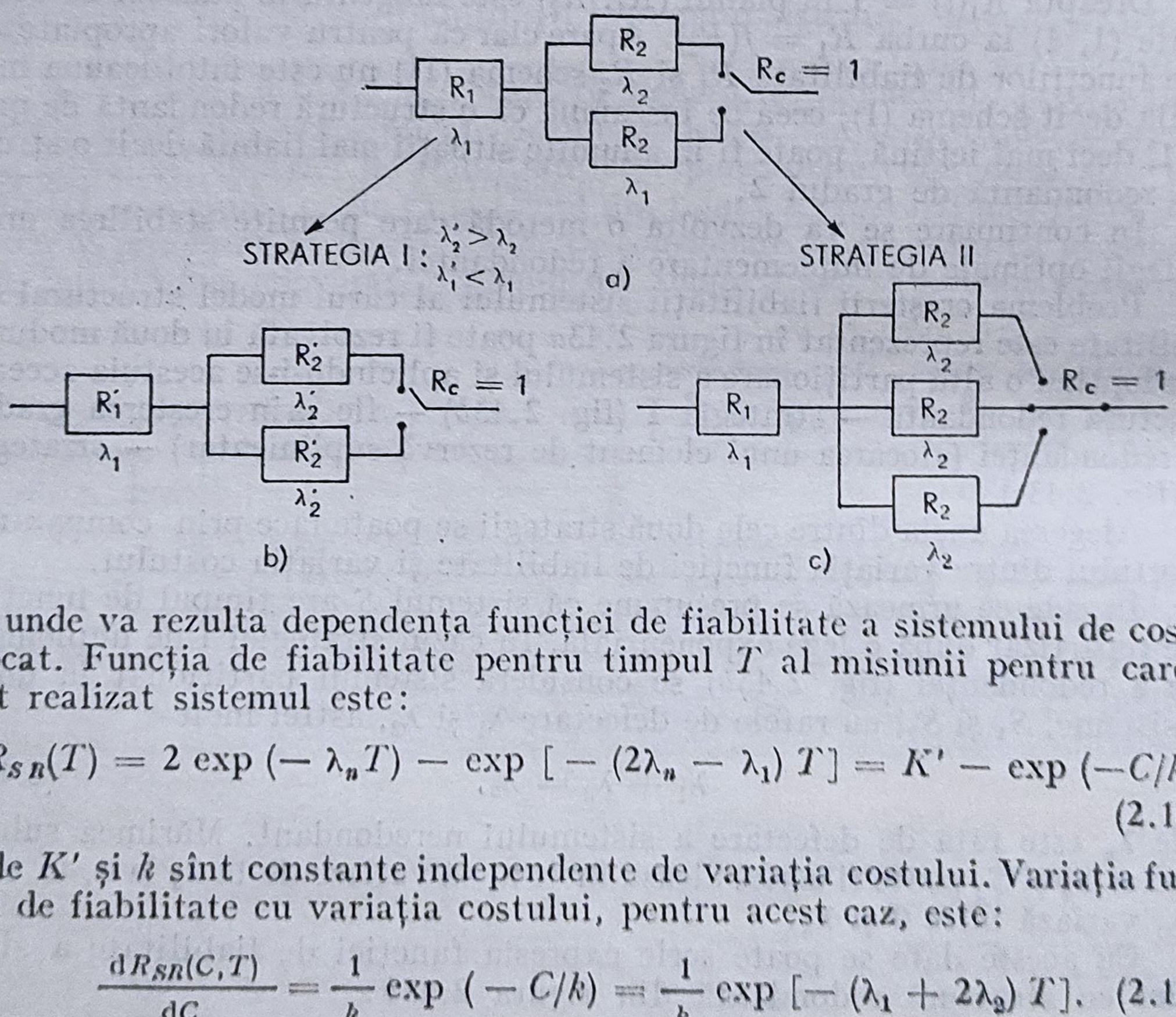


Fig. 2.43. Strategii de creștere a fiabilității:
a — sistemul redondant inițial ; *b* — altă partiționare dată sistemului pentru implementarea aceluiași tip de redondanță ; *c* — creșterea gradului redondanței.

de unde va rezulta dependența funcției de fiabilitate a sistemului de costul alocat. Funcția de fiabilitate pentru timpul T al misiunii pentru care a fost realizat sistemul este:

$$R_{Sn}(T) = 2\exp(-\lambda_n T) - \exp[-(2\lambda_n - \lambda_1)T] = K' - \exp(-C/k) \tag{2.108}$$

unde K' și k sînt constante independente de variația costului. Variația funcției de fiabilitate cu variația costului, pentru acest caz, este:

$$\frac{dR_{Sn}(C,T)}{dC} = \frac{1}{k}\exp(-C/k) = \frac{1}{k}\exp[-(\lambda_1 + 2\lambda_2)T]. \tag{2.109}$$

• În situația adoptării strategiei II de implementare a structurii redondante (fig. 2.43*c*), $\frac{dR_{Sn}}{dC}$ se calculează după cum urmează:

$$R_{Sn}(T) = \exp(-\lambda_1 T)\,[1 - (1 - \exp(-\lambda_2 T))^n]; \tag{2.110}$$

$$C = k(\lambda_1 + n\lambda_2)T; \tag{2.111}$$

$$\frac{dR_{SR}(T,n)}{dn} = -\exp(-\lambda_1 T)\,[1-\exp(-\lambda_2 T)]^n \cdot \ln\,[1-\exp(-\lambda_2 T)]; \tag{2.112}$$

$$\frac{dn}{dC} = \frac{1}{k\lambda_2 T}; \tag{2.113}$$

$$\frac{dR_{SR}(C)}{dC} = -\frac{\exp(-\lambda_1 T)}{k\lambda_2 T}\,[1-\exp(-\lambda_2 T)]^n.\ln\,[1-\exp(-\lambda_2 T)]. \tag{2.114}$$

Pentru a facilita alegerea uneia dintre strategii, se determină raportul variațiilor funcției de fiabilitate cu costul pentru cele două strategii. În cazul schemei considerate se obține:

$$\left.\frac{dR_{SR}(C,T)}{dC}\right|_{II} \Big/ \left.\frac{dR_{SR}(C,T)}{dC}\right|_{I} = -\frac{[1-\exp(-\lambda_2 T)]^3 \ln\,[1-\exp(-\lambda_2 T)]}{\lambda_2 T \exp(-2\lambda_2 T)}. \tag{2.115}$$

Atunci cînd raportul (2.115) este mai mare decît 1, este preferabilă strategia II, de alocare a unei rezerve suplimentare, rezultînd o creștere de fiabilitate superioară la aceeași creștere a costului. Atunci cînd raportul este mai mic decît 1, este preferabilă strategia I, de modificare a partiționării sistemului, întrucît în acest caz se produce o creștere mai mare a funcției de fiabilitate la variația costului.

O b s e r v a ț i e. Dacă raportul exprimat în relația (2.115), este egal cu unitatea, procedeul considerat nu mai permite luarea unei decizii privind alegerea uneia sau alteia dintre cele două strategii, motiv pentru care trebuie să se recurgă la un alt criteriu de comparație.

2.6.2. METODĂ DE STABILIRE A NIVELULUI OPTIM DE PARTIȚIONARE A UNUI SISTEM ÎN VEDEREA APLICĂRII REDONDANȚEI

După cum s-a arătat, redondanța poate fi aplicată global întregului sistem sau la nivel de subsisteme. Valoarea funcției de fiabilitate, ca de altfel și celelalte performanțe de fiabilitate, depind de gradul de partiționare al sistemului în subsistemele cărora li se aplică redondanța. În continuare se va prezenta o metodă de identificare a nivelului optim de partiționare a sistemului, astfel încît să se obțină o creștere a fiabilității de valoare maximă la același grad al redondanței. Pentru o cît mai completă edificare se vor analiza cîteva cazuri uzuale.

1.° *Cazul redondanței dinamice de gradul 1.* Se consideră o structură redondantă dinamică de tip activ, așa cum a fost definită în § 2.2.6. Comutatorul are funcția de fiabilitate $R(t)$, iar funcția de fiabilitate a sistemului neredondant căruia i se implementează redondanța este $R(t)^N$, N fiind raportul dintre numărul circuitelor electrice echivalente din punctul de vedere al funcției de fiabilitate ale sistemului neredondant și ale circuitului electric al comutatorului.

Sistemul căruia i se implementează această structură redondantă se partiționează în k subsisteme, cărora li se aplică tehnica de redondanță considerată. În continuare, se va demonstra că există o valoare a lui k, astfel încît prin utilizarea acestui procedeu să se obțină — pentru sistemul cu structură redondantă — o funcție de fiabilitate de valoare maximă.

Cu ipotezele enunțate mai înainte se poate scrie expresia funcției de fiabilitate pentru sistemul redondant considerat:

$$R_{SR}(t) = [(2R(t)^{N/k} - R(t)^{2N/k})\ R(t)]^k. \tag{2.116}$$

sau

$$R_{SR}(t) = R(t)^N\ [R\ (t)\ (2 - R(t)^{N/k})]^k. \tag{2.117}$$

Interesează dacă există un maxim al funcției $R_{SR}(t)$ la variația lui k. Pentru aceasta — pe baza procedurii clasice din analiza matematică — se studiază semnul derivatei funcției $R_{SR}(t)$ în raport cu k:

$$\frac{\partial R_{SR}(t,k)}{\partial k} = R(t)^N \frac{\partial}{\partial k}\left\{[R(t)(2 - R(t)^{N/k})]^k\right\}; \tag{2.118}$$

$$\operatorname{sgn}\frac{\partial R_{SR}(t,k)}{\partial k} = \operatorname{sgn}\frac{\partial}{\partial k}\left\{[R(t)(2 - R(t)^{N/k})]^k\right\} =$$

$$= \operatorname{sgn}\frac{\partial}{\partial k}\{[\varphi(k)]^k\}. \tag{2.119}$$

$$\text{Dar } \frac{\partial}{\partial k}[\varphi(k)]^k = k\varphi'(k)\ [\varphi(k)]^{k-1} + [\varphi(k)]^k \ln\varphi(k),$$

iar $[\varphi(k)]^k > 0$ pentru $\forall k$; avînd în vedere aceste relații rezultă:

$$\operatorname{sgn}\frac{\partial}{\partial k}\left\{[\overline{R(t)}(\overline{2 - R(t)^{N/k}})]^k\right\} = \operatorname{sgn}\left\{R(t)^{N/k} \cdot \frac{N}{k} \ln R(t) + \right.$$

$$\left. + (2 - R(t)^{N/k}) \ln[R(t)(\overline{2 - R(t)^{N/k}})]\right\}. \tag{2.120}$$

Pentru gradul maxim de partiționare a sistemului, $k = N$, semnul derivatei $\frac{\partial R_{SR}(t,k)}{\partial k}$ este:

$$\operatorname{sgn}\frac{\partial R_{SR}(t,k)}{\partial k} = \operatorname{sgn}\{R(t) \ln R(t) + (2 - R(t)) \ln [R(t)(2 - R(t))]\} =$$

$$= \operatorname{sgn}[2 \ln R(t) + (2 - R(t)) \ln(2 - R(t))]. \tag{2.121}$$

Problema s-a redus astfel la studierea semnului funcției

$$\psi(t) = 2 \ln R(t) + (2 - R(t))\ \ln(2 - R(t)). \tag{2.122}$$

Se constată cu ușurință că $\frac{d\psi(t)}{dR(t)} > 0$ pentru $0 < R(t) < 1$, de unde rezultă că funcția ψ, continuă pe domeniul de definiție al lui R, este

crescătoare cu creşterea lui R. Analiza variaţiei funcţiei ψ cu R este concentrată în tabelul (2.22).

Tabelul 2.22. VARIAŢIA FUNCŢIEI $\psi(t) = 2 \ln R(t) + (2 - R(t)) \ln (2 - R(t))$ CU VALORILE LUI $R(t)$

$R(t)$	0 1
$\frac{d\psi(t)}{dR(t)}$	+ + +
$\psi(t)$	∞ ↖ 0

Rezultă că $\operatorname{sgn} \frac{\partial R_{SR}(t,k)}{\partial k} < 0$ pentru $k = N$, atunci cînd funcţia de fiabilitate $R(t)$ ia valori între 0 şi 1.

Pentru $k = 1$, gradul minim de partiţionare a sistemului, se obţine:

$$\operatorname{sgn} \frac{\partial R_{SR}(t,k)}{\partial k}\Big|_{k=1} = \operatorname{sgn} \left\{ R(t)^{N/k} \cdot \frac{N}{k} \ln R(t) + (2 - R(t)^{N/k}) \times \right.$$

$$\left. \times \ln [R(t)(2 - R(t))^{N/k}] \right\}_{k \to 1} = \operatorname{sgn} [2 \ln (2\ R(t))]. \qquad (2.123)$$

Valorile uzuale ale lui $R(t)$ sînt mai mari decît 0,5, aşa încît, pentru $k = 1$,

$$\operatorname{sgn} \frac{\partial R_{SR}(t,k)}{\partial k} > 0$$

la variaţia funcţiei de fiabilitate $R(t)$ între 0,5 şi 1.

Deoarece $\frac{\partial R_{SR}(t,k)}{\partial k}$ este o funcţie continuă în raport cu k pentru $k \in [1,N]$, rezultă că există cel puţin un k_0 pentru care $\frac{\partial R_{SR}(t,k)}{\partial k}\Big|_{k_0} = 0$, atunci cînd $R(t)$ ia valori între 0 şi 1. Deci, funcţia de fiabilitate a sistemului redondant considerat va avea un extrem — un maxim — la variaţia gradului partiţionării sistemului, în vederea asocierii unei structuri redondante la fiecare partiţie.

În continuare se va arăta că această valoare este unică, în care scop, se calculează:

$$\frac{\frac{\partial R_{SR}(t,k)}{\partial k}}{R_{SR}(t,k)} = \frac{\frac{\partial}{\partial k} [R(t)(2 - R(t)^{N/k})]^k}{[R(t)\ (2 - R(t)^{N/k})]^k} =$$

$$= \frac{N}{k} \frac{R(t)^{N/k} \ln R(t)}{(2 - R(t)^{N/k})} + \ln R(t)(2 - R(t)^{N/k}). \qquad (2.124)$$

Se notează $R(t)^{N/k} = u$; atunci cînd $1 < k < N$, variabila u va satisface inegalitatea $0 < u < 1$.

Rezultă că valoarea k_0 căutată va fi dată de ecuaţia:

$$\frac{\frac{\partial R_{SR}(t,k)}{\partial k}}{R_{SR}(t,k)} = \frac{u \ln(u)}{2 - u} + \ln [R(t)(2 - u)] = 0. \qquad (2.125)$$

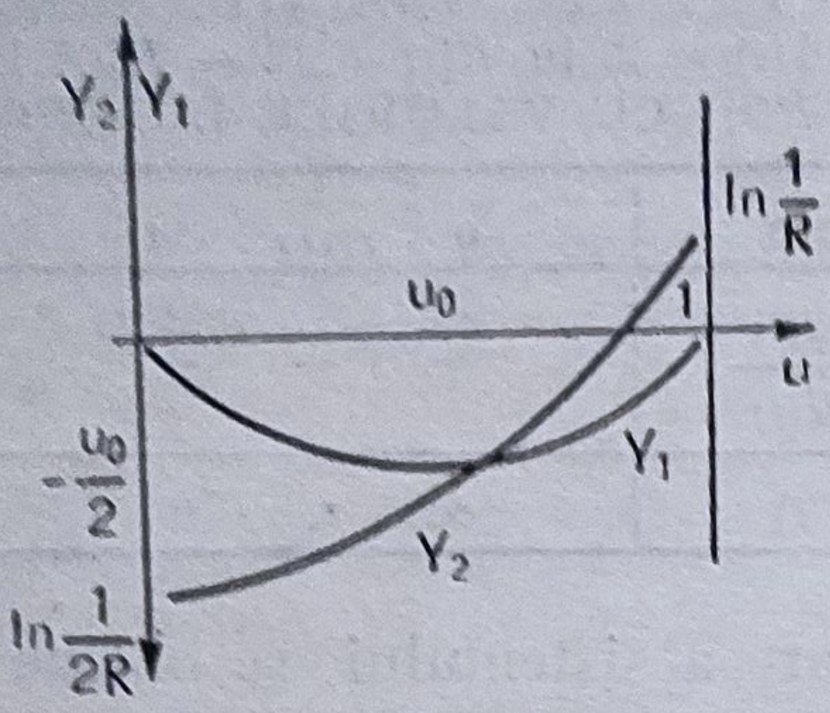


Fig. 2.44. Identificarea gradului optim de partiționare.

Se notează:

$$\frac{u \ln u}{2-u} = y_1 \quad \text{și} \quad ln[R(2-u)] = -Y_2, \tag{2.126}$$

astfel încît rezolvarea ecuației (2.125) se reduce la găsirea intersecției graficelor funcțiilor Y_1 și Y_2.

Analiza funcției continue Y_1 la variația variabilei u între 0 și 1 este indicată în tabelul 2.23, iar în tabelul 2.24 se prezintă variația funcției Y_2.

În figura 2.44 sînt trasate graficele funcțiilor Y_1 și Y_2, care se intersectează într-un singur punct; deci, există o singură valoare a lui k pentru care $\frac{\partial R_{SR}}{\partial k} \Big/ R_{SR}$ se anulează.

Tabelul 2.23. VARIAȚIA FUNCȚIEI $y_1 = \frac{u \ln u}{2-u}$ CU VALORILE LUI $u \in [0,1]$.

u	0	u_0	1
$\frac{dy_1}{du} = \frac{2 \ln u + 2 - u}{(2-u)^2}$	$-\infty$	$-$	0	$+$	$+1$
$y_1 = \frac{u \ln u}{2-u}$	0	$\searrow$	$-\frac{u_0}{2}$	$\nearrow$	0

Tabelul 2.24. VARIAȚIA FUNCȚIEI $y_2(u) = -\ln [R(2-u)]$

u	0		1
$\frac{dy_2}{du} = \frac{1}{2-u}$	$\frac{1}{2}$	$+$	$+1$
y_2	$\ln \frac{1}{2R}$	$\nearrow$	$\ln \frac{1}{R}$

Pornind de la aceste dezvoltări se poate stabili o metodă de calcul al valorii optime pentru gradul de partiție a sistemului în vederea implementării redondanței, metodă ce cuprinde următoarele etape:

- se calculează derivata funcției de fiabilitate a sistemului redondant în raport cu k, numărul de module funcționale ale acestuia cărora li se implementează redondanța;

• se alege valoarea întreagă cuprinsă între două valori ale lui k, pentru care funcția $\frac{\partial R_{SR}}{\partial k}$ are semne contrare, ce va corespunde punctului de maxim al funcției de fiabilitate.

O b s e r v a ț i e . Din analiza relațiilor (2.121) ÷ (2.123) rezultă că valoarea optimală a lui k este dependentă de funcția de fiabilitate a sistemului căruia i se implementează acest tip de redondanță și deci, implicit, de timpul necesar îndeplinirii misiunii.

2°. *Cazul structurii redondante dinamice de gradul* 2. În acest caz comutatorul are trei poziții de lucru; evident, structura sa este mai complicată decît cea a comutatorului din cazul 1°; ca urmare el va conține mai multe elemente. În scopul comparării rezultatelor corespunzătoare situației analizate cu cele obținute în cazul 1°, se adoptă ca element de referință funcția de fiabilitate a comutatorului structurii redondante de comutație de gradul 1. Funcția de fiabilitate a comutatorului utilizat în acest caz este $R(t)^2$, deoarece numărul componentelor va fi practic dublat. Deci, funcția de fiabilitate a sistemului partiționat în k module funcționale pentru a i se implementa acest tip de redondanță este:

$$R_{SR}(t,k) = [(R(t)^{3N/k} - 3R(t)^{2N/k} + 3R(t)^{N/k})\, R^2(t)]^k = \\ = R(t)^N[(R(t)^{2N/k} - 3R(t)^{N/k} + 3)\, R^2(t)]^k . \qquad (2.127)$$

În mod analog se demonstrează că există un singur maxim al funcției de fiabilitate $R_{SR}(t,k)$ la variația gradului de partiționare a sistemului, în același mod procedîndu-se și pentru determinarea valorii optime a parametrului k.

3°. *Cazul unei structuri redondante de tip logică majoritară*. Se consideră că funcția de fiabilitate a voterului este $R(t)$, iar a sistemului neredondant căruia i se implementează această structură, $R(t)^N$.

Funcția de fiabilitate a sistemului partiționat în k subsisteme cărora li se implementează structura redondantă logică majoritară de tip 2 din 3 este:

$$R_{SR}(t,k) = [(3R(t)^{2N/k} - 2R(t)^{3N/k})\, R(t)]^k = \\ = R(t)^{2N}[(3 - 2R(t)^{N/k})\, R(t)]^k . \qquad (2.128)$$

În scopul determinării valorii optime a gradului de partiționare se procedează ca în cazul 1°.

Pentru situația ideală, cînd comutatorul structurii redondante de comutație sau voterul structurii redondante logică majoritară are valoarea funcției de fiabilitate egală cu 1, se demonstrează [17] că functia de fiabilitate a sistemului redondant tinde la 1, atunci cînd gradul de partiționare a sistemului tinde la infinit.

2.6.3. CONCLUZII

În § 2.6 au fost reliefate cîteva probleme implicate în determinarea nivelului optim de implementare a unei structuri redondante.

Pe baza unor studii de caz au fost indicate cîteva procedee de stabilire a strategiilor optimale de implementare a redondanței. A fost reliefată necesitatea efectuării în faza de proiectare a unor analize de tipul celor indicate, anterior, în scopul stabilirii gradului optim de aplicare a redondanței, sau a modificării partiționării sistemului astfel încît să se obțină o fiabilitate maximă.

S-a demonstrat că există un grad optim de partiționare a sistemului, corespunzător tipului redondanței implementat și unui timp dat al misiunii sistemului.

Atît în stabilirea unei strategii de implementare a unei structuri redondante, cît și pentru găsirea nivelului de partiționare a unui sistem în vederea implementării redondanței, s-a urmărit obținerea unei funcții de fiabilitate de valoare maximă. Realizarea unui maxim în fiabilitate prin adoptarea unei anumite structuri redondante nu este întotdeauna soluția optimă adoptată în practică, dar în cele mai multe cazuri este soluția cea mai acceptabilă.

O problemă care trebuie avută în vedere la partiționarea sistemului este aceea a minimizării nodurilor structurii redondante, care implică o serie de cerințe pentru sistemele de decizie in ceea ce privește detecția erorilor, mecanismele de comutație etc. Aceste cerințe apar dificil de implementat în tehnologie LSI sau VLSI în primul rînd din cauza raportului poartă/pin, care nu trebuie să fie mare, iar nodurile trebuie minimizate pentru că introduc ele înseși mecanisme de defectare. Dar, pe de altă parte, aplicarea redondanței la nivelul întregului sistem, global, nu conduce întotdeauna la rezultate satisfăcătoare [5], [16], [46].

BIBLIOGRAFIE

1. ABRAHAM, I.A.; SIEWIOREK, D.P., *An Algorithm for the Accurate Reliability Evaluation of Triple Modular Redundancy Networks*, IEEE Trans. on Computers, C-23, 7, 1974, p. 682—692.
2. AVIZIENIS, A., *Design of Fault Tolerant Computers*, AFIPS Conf. Proc., 31, Washington, 1967, p. 733—743.
3. AVIZIENIS, A.; GILLEY, G.G.; MATHUR, F.P.; RENNELS, D.A.; ROHR, J.A.; RODIN D.K., *The STAR Computer : An Investigation of the Theory and Practice of Fault Tolerant Computer Design*, IEEE Trans. on Computers, C-20, 1, 1971.
4. BACIVAROF, ANGELICA, *Generator de tact de înaltă fiabilitate, Cercetări în tehnologie electronică și fiabilitate*, Editura Didactică și Pedagogică, București, 1981, p. 260—264.
5. BACIVAROF, ANGELICA, *Sisteme digitale tolerante la defectări, Teză de doctorat*, Institutul Politehnic, București, 1980.
6. BACIVAROF, ANGELICA, *Model pentru analiza performanțelor de fiabilitate ale unui automat cu structură redondantă*, Buletinul Institutului Politehnic, XLVIII, București, 1986, p. 73—79.
7. BACIVAROF, I.C., BACIVAROF, ANGELICA, *Evaluarea fiabilității circuitelor integrate*, Revista Transporturilor și Telecomunicațiilor, 3, 1979, p. 179—185.
8. BACIVAROF, I.C., *Contribuții la studiul fiabilității unor sisteme de telecomunicații, Teză de doctorat*, Institutul Politehnic, București, 1979.
9. BAZOVSKY, J., *Fiabilité. Théorie et pratique de la sûreté de fonctionnement*, Dunod, Paris, 1966.
10. CARTER, W.C., *Design Techniques for Modular Architecture for Reliable Computer Systems*, IBM. Res. Rep. RA12, T.J. Watson Res. Cent., 1979.
11. CARTER, W.C.; BOURICIUS, W.G., *A Survey of Fault Tolerant Architecture and Its Evaluation*, Technical Report, RC 3154, IBM Res. Cent, N.Y., 1981.
12. CARTER, W.C., *A Theory of Design of Fault-Tolerant Computers Using Standby Sparing*, Digest of the Int. Symp. on Fault Tolerant Computers, 1971, p. 83—86.

13. CARTER, W.C.; Mc. CARTHY, C.E., *Implementation of an Experimental Fault Tolerant Memory System*. IEEE Trans. on Computers, C-25, 10, 1976, p. 557—568.
14. CĂTUNEANU, V.M., ş.a., *Materiale pentru electronică. Teoria fiabilităţii şi control statistic*, Editura Didactică şi Pedagogică, Bucureşti, 1983.
15. CĂTUNEANU, V.M., BACIVAROF, I.C., *Fiabilitatea sistemelor de telecomunicaţii*, Editura Militară, Bucureşti, 1985.
16. COURTOIS, B., *Safety — Availability and Maintenance Trade — Offs for Logical Systems*, Technical Report, Grenoble, 1978.
17. COURTOIS, B., *Etude d'un calculateur tolerant de pannes*. Inst. National Polytechnique de Grenoble, 1976.
18. DAVIES, D.; WAKERLEY, I., *Synchronization and Matching in Redundant Systems*, IEEE Trans. on Computers, C-27, 6, 1978, p. 531—539.
19. FREEMAN, H.A.; METZE, G., *Fault Tolerant Computers Using „Dotted Logic" Redundancy Techniques*, IEEE Trans. on Computers, C-21, 8, 1972, p. 867—871.
20. GEBER, T.; STĂICUŢ, E.; TUTOVEANU, I.; POPA, I.; GRIGORESCU, M., *Fiabilitatea şi mentabilitatea sistemelor de calcul*, Editura Tehnică, Bucureşti, 1982.
21. GIRARD, E.; RAULT, J.C.; TULLONE, R., *Les méthodes probabilistes de generation des sequences de test : estimation de l'eficacité et de la longueur des sequences*, Rev. techn. Thomson CSF, 1, 1974.
22. GOLDENBERG, L.M., *Teoria i rascet impulsionîh ustroistv na poluprovodnikovîh priborah*, Izd. Sviazi, Moskva, 1969.
23. GULLMAN, G., *Coduri detectoare şi corectoare de erori*, Editura Tehnică, Bucureşti, 1972.
24. GUNNINGBERG, P., *Fault — Tolerance Implemented by Voting Protocols in Distributed Systems*, Institute of Technology, Uppsala Univ. Uppsala, 1985.
25. HAMMING, R.W., *Error Detecting and Error Corecting*, Bell Syst. Techn. Journal, 26, 1950.
26. HAMPEL, D.; WINDER, R.O., *Threshold Logic*, IEEE Spectrum, may, 1971, p. 32—39.
27. HOPKINS, A., *FTMP — A Highly Reliable Fault Tolerant Multiprocesor for Aircraft*, Proc. of IEEE, 66, 10, 1978, p. 1 221—1 239.
28. KLASCHKA, THERESA, *A Method for Redundancy Scheme Performance Asessment*, IEEE Trans. on Computers, C-20, 11, 1971, p. 1 371—1 376.
29. KLASCHKA, THERESA, *Reliability, Improvement by Redundancy in Electronic Systems*, Technical Report, 72 200, Royal Aircraft Establishment, 1973.
30. LAPRIE, J.C., *On Reliability Prediction of Repairable Redundant Digital Structures*, IEEE Trans. on Reliability, R-25, 3, 1976.
31. LAPRIE, J.C.; COSTES, A., *Dependability : An Unifying Concept for Reliable Computing*, Proc. 12 th Int. Symp. on Fault-Tolerant Computing, Los Angels, 1982, p. 18—21.
32. LARSEN, R.W.; REED, R.S., *Redundancy by Coding versus Redundancy by Replication*, IEEE Trans. on Computers, C-21, 2, 1972.
33. MATHUR, F.P.; AVIZIENIS, A., *Reliability Analysis and Architecture of a Hybrid Redundant Digital System. Generalized Triple Modular Redundancy with Self-Repais*, AFIPS Conf. Proc., 36, 1970, p. 375—383.
34. MATHUR, F.P., *On Reliability and Analysis of Ultrareliable Fault-Tolerant Digital Systems*, IEEE Trans. on Computers, C-20, 11, 1971, p. 1 376—1 382.
35. Mc CONNEL, S.; SIEWIOREK, D., *Synchronization and Voting*, IEEE Trans. on Computers, C-30, 2, 1981, p. 161—164.
36. MOORE, E.F.; SHANNON, C.E., *Reliable Circuits Using Less Reliable Relays*, J. Franklin Inst., 1956, p. 191—208, 281—297.
37. MORGAN, D.E.; TAYLOR, D.J.; CUSTEAU, G., *A Survey of Methods for Improving Computer Network Reliability and Availability*, Computer, 10, 1977, p. 42—50.
38. MUNTEAN, F., *Sinteza automatelor finite*, Editura Tehnică, Bucureşti, 1972.
39. Von NEUMAN, J., *Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components*, Annals of Mathematical Studies, 34, Princeton Univ. Press, 1956, p. 43—98.
40. OKONSKI, E.S., *Reliability of Digital Systems*, State Univ. of New York at Buffalo, New York, 1973.
41. OSAKI, S.; NISHIO, T., *Reliability Evaluation of Some Fault-Tolerant Computer Architectures*, Springer-Verlag, Berlin, 1986.
42. PARHAMI, B., *Interconection Redundancy for Reliability Enhancement in Fault-Tolerant Digital Systems*, Digital Processes, 5, 3—4, 1979, p. 199—211.

43. PIERCE, W.H., *Failure — Tolerant Computer Design*, Academic Press, New York, 1965.
44. PILAUD, E., *Conception et validation de systèmes informatiques à haute sureté de fonctionnement*, Inst. National Polytechnique de Grenoble, 1984.
45. RANDELL, B.; LEE, P.A.; TRELLAVEN, P.C., *Reliability Issues in Computing System Design*, A C M Comput. Surv., **10**, 1978, p. 123—165.
46. RENNELS, D.A., *Architectures for Fault Tolerant Spacecraft Computers*, Proc. of the IEEE, **66**, 1978, p. 1 255—1 268.
47. RENNELS, D., *Distributed Fault-Tolerant Computer Systems*, Computer, **13**, **3**, 1980, p. 55—65.
48. SCHMITT, E., *Information Processing Systems*, Technical Report, EE 570, State Univ. of New York at Buffalo, N.Y., 1972.
49. SCHWOB, M., PEYRACHE, G., *Traité de fiabilité*, Masson, Paris, 1969.
50. SHERIDAN, C., *Space Shuttle Software*, Datamation, **7**, 1978, p. 128—140.
51. SHOOMAN, M., *Probabilistic Reliability, An Engineering Approach*, Mc Graw-Huill, New York, 1968.
52. SIEWIOREK, D.P., *A Measure of Switch Complexity in Systems with Standby Sparing*, Technical Report, Stanford Univ., Stanford, 1974.
53. SIEWIOREK, D.P.; Mc CLUSKEY, E.J., *An Iterative Cell Switch Design for Hybrid Redundancy*, IEEE Trans. on Computers, **C-22**, **3**, 1973, p. 290—297.
54. SIEWIOREK, D.P., *Reliability Modeling of Compensating Module Failures in Majority Voted Redundancy*, IEEE Trans. on Computers, **C-24**, **5**, 1975, p. 525—533.
55. SIEWIOREK, D.P., *A Case Study of Cmmp, Cm and Cvmp :* Part. I — *Experiences with Fault-Tolerance in Multiprocessor Systems ;* Part. II — *Predicting and Calibrating Reliability of Multiprocessor Systems*, Proc. of IEEE, **66**, **10**, 1978, p. 1 160—1 200.
56. SKLAROFF, J., *Redundancy Management Technique for Space Shuttle Computers*, IBM J. Res. Devel., **20**, **1**, 1976, p. 20—28.
57. STAS 10 307-75, *Fiabilitatea produselor industriale. Indicatori de fiabilitate.*
58. TEOSTE, R.; *Digital Circuit Redundancy*, IEEE Trans. on Reliability, **R-13**, 2, 1964, p. 42—61.
59. THEURETZBACHER, N., *VOTRICS: Voting Triple Modular Computing System*, Digest of Papers 16-th Int. Symp. on Fault-Tolerant Computing, 1986, p. 144—151.
60. TRYON, J.G., *Quadded Logic, Redundancy Techniques for Computing Systems*, Spartan Books, Washington, 1962, p. 205—228.
61. WAKERLEY, J., *Microcomputer Reliability Improvement Using Triple Modular Redundancy*, Proc. of IEEE, **64**, *6*, 1976.
62. WENSLEY, J., *SIFT : Design and Analysis of a Fault Tolerant Computer, for Aircraft Control*, Proc. of IEEE, **66**, **10**, 1978, p. 1 240—1 255.
63. * * * *Special Issues on Fault-Tolerant Computers*, IEEE Transactions on Computers, 1971—1987.

CAPITOLUL 3

TEHNICI DE DETECȚIE A ERORILOR

3.1. CONSIDERAȚII ASUPRA TEHNICILOR DE DETECȚIE A ERORILOR ÎN VEDEREA IMPLEMENTĂRII TOLERANȚEI LA DEFECTĂRI

3.1.1. INTRODUCERE

Punctul de pornire al strategiilor de tolerare a defectărilor îl constituie detecția stării eronate a sistemului, stare care în absența oricăror măsuri de protecție va conduce la defectarea acestuia. Mai mult, succesul oricăreia dintre aceste strategii este dependent, în mod critic, de eficacitatea acestei detecții.

În principiu, evidențierea stării eronate a unui sistem — stare care se datorează fie erorilor de proiectare, fie defectării elementelor componente — se realizează prin introducerea în sistem a unor componente suplimentare (elemente hardware sau programe specifice), urmînd ca, ulterior, celelalte faze ale toleranței la defectări să mascheze aceste erori sau defecte fizice.

În general, identificarea stărilor eronate sau a defectelor fizice — diagnosticarea defectărilor sistemului —, împreună cu mentenanța corectivă sînt metode eficiente de asigurare a bunei funcționări a sistemelor mai ales în cazul proceselor *off-lne*, deoarece este posibilă reprocesarea funcțiunilor după detectarea si inlăturarea defectelor fără urmări prea grave, dar ele pot fi utilizate și in cazul proceselor *on-line*, de exemplu, ca modalități de a asigura operarea sistemelor cu redondanță de comutație.

Dacă se consideră evoluția în timp a procesului de diagnoză tehnică, în particular pentru sistemele de calcul, se constată că, pentru primele calculatoare, tehnicienii de înaltă calificare puteau ca pe baza experienței lor „clinice" să diagnosticheze și să înlăture defectările; datorită complexității sistemelor de calcul actuale este însă dificil să se acumuleze o astfel de experiență.

Pentru sistemele digitale actuale procedura de mentenanță se bazează pe utilizarea unui ansamblu de module localizate în exteriorul sistemului, sau integrate în structura sa. Aceste unități suplimentare asigură dialogul cu operatorul și aplică într-un mod semiautomat sau automat secvențele testului de diagnostic. Această abordare este utilă deoarece, cu prețul utilizării unui număr restrîns de structuri suplimentare, mentenanța este rapidă și eficace.

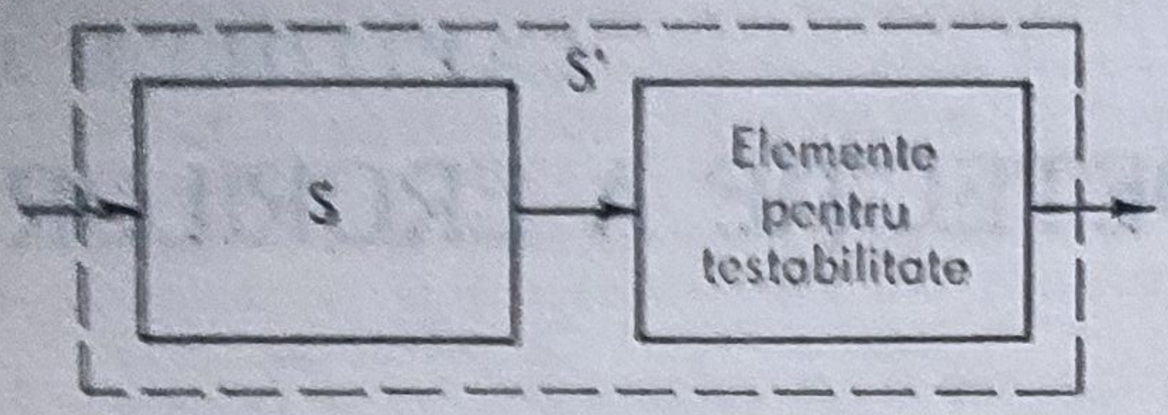


Fig. 3.1. Structura unui sistem testabil.

Se consideră un sistem *S*, proiectat pentru realizarea unei anumite funcțiuni. Atunci cînd domeniul specificațiilor $\mathcal{S}_p$ nu coincide cu cel indicat, *S* este considerat defect. Rezultă că implementarea toleranței la defectări, în cazul acestui sistem, presupune detecția erorilor, bazată pe „interceptarea" ieșirilor sistemului *S* și testarea lor în conformitate cu specificațiile, urmată de introducerea unor tehnici de tolerare a defectărilor identificate. Se obține astfel un nou sistem, *S'* (fig. 3.1), care are în structura sa atît sistemul *S*, cît și elemente suplimentare care realizează testarea bunei funcționări a sistemului *S*. Aceste elemente suplimentare realizează funcția τ' de diagnosticare a defectărilor sistemului *S* (§ 1.4.2), sau doar identificarea erorilor în semnalele de la ieșirea sistemului *S*.

3.1.2. CONCEPTUL DE TESTARE A FUNCȚIONĂRII

Conceptul de testare a funcționării unui sistem acoperă de fapt mai multe probleme, legate între ele, dar care prezintă grade diferite de interes, funcție de nivelul la care are loc testarea: la nivel de circuit — în cazul sistemelor electronice — sau la nivelul global al unui sistem.

Prima problemă este aceea a *detecției defectărilor*, care semnifică identificarea prezenței sau absenței unei defectări în sistem.

Se consideră un sistem cu n intrări și m ieșiri. Fiecare semnal de ieșire este o funcție logică de semnale de intrare:

$$y_i = f_i(x_1, x_2 \ldots, x_n), \quad i = 1,2,\ldots, m. \tag{3.1}$$

Fie *N* mulțimea celor 2^n combinații posibile ale celor n semnale de intrare, în cazul semnalelor binare. Dacă oricare ar fi i, cele m identități (3.1) sînt verificate de combinațiile cuprinse în mulțimea *N*, se consideră că sistemul nu are nici o defectare; din contră, dacă cel puțin o identitate nu este verificată, atunci sistemul funcționează defectuos.

Deci, problema detecției defectărilor poate fi formulată sub forma următoare:

Să se găsească o submulțime de combinații ale semnalelor de intrare *N'* din mulțimea *N*, astfel încît aplicarea ei pe *intrările primare* ale sistemului permite — prin observarea răspunsului la ieșiri — evidențierea prezenței sau absenței defectărilor. Combinația *N'* a semnalelor de intrare va fi numită mulțimea semnalelor de detecție. Dacă numărul elementelor submulțimii *N'* este minimal, secvența de test corespunzătoare se va numi minimală. O secvență de test este completă dacă aplicarea sa permite detecția tuturor defectărilor din sistem; în caz contrar secvența va fi incompletă.

Cunoașterea unei secvențe minimale are avantajul de a conduce la timpi de test foarte scurți, dar cercetările pentru găsirea unei astfel de secvențe pot fi îndelungate. De aceea — în general — se acceptă un compromis între timpii necesari testării și timpii necesari găsirii secvențelor de test corespunzătoare.

A doua problemă a testării privește localizarea sau diagnosticarea defectărilor. Odată ce o funcționare defectuoasă a apărut este de dorit identificarea cauzei: subsistem al sistemului sau element al subsistemului. Astfel localizarea poate fi efectuată fie la nivelul unei componente, fie la nivelul unei mulțimi de componente, ceea ce permite identificarea și implicit înlocuirea componentei defecte sau a celui mai mic subansamblu care o conține. Testele de acest tip, necesare în mod frecvent în realizarea toleranței la defectări sau în operațiile de mentenanță corectivă, trebuie să fie cît mai concise și cît mai rapide, mai ales cînd se folosesc în faza de testare inițială. Sînt de obicei teste de tipul funcționare-nefuncționare (*go no go*).

Problema localizării unei defectări sau a diagnosticării acesteia poate fi formulată în forma următoare:

Fie P mulțimea celor n defectări susceptibile de a afecta sistemul. Fiecărei defectări d_i îi corespunde o submulțime N_{d_i} din mulțimea semnalelor N, numită secvența semnalelor de detecție. Fie N_d mulțimea semnalelor de detecție: relația dintre o defectare și o submulțime de detecție definește o aplicație a lui P în N_d. În cazul în care această aplicație este injectivă, submulțimile N_{d_i} sînt distincte și defectările se pot discerne.

În practică acest deziderat nu este — în general — îndeplinit, deoarece funcționarea defectuoasă a unui element/sistem nu poate fi atribuită — de obicei — unei singure defectări, ci mai multora; astfel, generarea secvențelor de diagnostic constă în împărțirea mulțimii secvențelor N_d în submulțimi N_{d_i} precum și în stabilirea unei strategii de derulare a testelor pentru a putea discerne fiecare element al sistemului cercetat. În acest context apare și problema minimizării lungimii secvențelor de test, care poate fi soluționată în parte prin găsirea unui compromis între lungimea testului și nivelul de diagnostic dorit.

Avîndu-se în vedere tipul acțiunii vizate — detecție, reglaj, reparare — testele aplicate sistemelor se pot clasifica în următoarele categorii:

1°. *Teste pentru verificarea funcționării*, orientate spre o verificare de conformitate cu specificarea funcțională a sistemului;

2°. *Teste de încredere*, pentru verificarea diferitelor funcții ale sistemelor în scopul obținerii datelor asupra stării sistemului. Aceste teste sînt utilizate de regulă după o intervenție corectivă în sistem sau periodic în cadrul programului de mentenanță preventivă;

3°. *Teste de diagnostic*, în vederea operațiilor de înlocuire sau reglare necesare, după detectarea unei defectări a sistemului.

Generarea acestor teste este bazată pe o structurare și ordonanțare riguroasă a hardware-ului sistemului.

3.1.3. PARTICULARITĂȚI ALE TESTĂRII SISTEMELOR ÎN VEDEREA IMPLEMENTĂRII TOLERANȚEI LA DEFECTĂRI

Pentru realizarea în bune condiții a unei testări, elementele suplimentare introduse pentru asigurarea testabilității sistemului (fig. 3.1) trebuie să posede următoarele caracteristici:

(1) să se bazeze pe specificaţiile sistemului;

(2) să testeze toate specificaţiile posibile ale sistemului, pentru a se realiza o testare completă a acestuia;

(3) defectările sistemului să nu conducă la defectarea elementelor care testează (testerului).

În realitate, cele mai multe sisteme nu îndeplinesc toate aceste trei condiţii. Pentru sistemele mari, de exemplu, nu este posibil să se realizeze o testare completă a tuturor performanţelor, care de altfel nici nu sînt specificate în totalitate, astfel încît primele două criterii nu pot fi în general realizate. De asemenea, nici independenţa testerului faţă de subsistemul testat nu poate fi realizată în totalitate în practică. Mai întîi, pentru că testerul trebuie să aibă acces la informaţia de testat şi prin aceasta o poate altera; în al doilea rînd, pentru că implementarea oricărui tester trebuie să fie bazată pe considerarea structurii sistemului şi a tipului de defectări avut în vedere.

Rezultă deci că problema realizării unor testere ideale pentru detecţia sau identificarea erorilor unui sistem nu este abordabilă în practică. Pentru cele mai multe sisteme, mai ales pentru cele care realizează toleranţa la defectări, se admite o testare a *acceptabilităţii* domeniului de funcţionare. Testele de acceptabilitate au în vedere un număr de defectări anticipate ale sistemului, ceea ce face ca aceste teste să fie în general mai simple şi mai eficiente decît un test general al operării sistemului. De asemenea, se admite că în cazul sistemelor mari, formate din mai multe procese concurente, o testare a fiecărui proces în parte este mai eficientă decît testarea stării întregului sistem.

În cazul sistemelor tolerante la defectări detecţia stărilor eronate se poate face în două moduri:

(a) aplicîndu-se sistemului S un test după generarea semnalelor de ieşire, dar înainte ca acestea să fie evidenţiate la ieşirea sistemului S' (fig. 3.1), urmînd ca apoi să se aplice una din tehnicile de tolerare a defectărilor;

(b) dîndu-se sistemului o structură de tip autotestabil care va însemna o testare continuă în timpul funcţionării normale a acestuia; în acest caz, vor fi înlăturate activităţile sistemului scurse între tranziţia eronată şi aplicarea testului, rezultînd totodată un timp mai redus de restabilire a funcţionării sistemului.

Aceste teste se pot clasifica în următoarele categorii, avînd în vedere principiile care stau la baza generării lor: teste sincrone; teste de diagnostic; teste realizate prin multiplicarea activităţii sistemului; teste realizate prin inversarea funcţiilor sistemului; teste realizate prin codarea adecvată a funcţiilor / stărilor sistemului.

Testele sincrone pot fi aplicate atît sistemelor hardware cît şi celor software [10]. De exemplu, ele sînt utilizate pentru detecţia erorilor în sistemul tolerant la defectări TANDEM 16 [47]. La fiecare secundă fiecare procesor trimite un mesaj pentru testare celorlalte procesoare, iar la fiecare două secunde este realizat un test de verificare a mesajelor transmise spre celelalte procesoare. Dacă un mesaj nu a fost recepţionat, atunci procesorul corespunzător este considerat defect şi sînt semnalate o serie de stări de „excepţie".

Testele de diagnostic sînt caracteristice testării domeniului specificaţiilor elementelor componente ale sistemului şi nu domeniului specificaţiilor sistemului. Un test de diagnostic implică excitarea unei componente cu un set de semnale la intrări pentru care sînt cunoscute semnalele de ieşire corecte. Semnalele de ieşire obţinute sînt comparate cu valorile aşteptate, iar apariţia unei discrepanţe indică prezenţa unui defect în componenta testată.

Privite sub raportul timpului și al resurselor cerute pentru execuția lor, testele de diagnostic sînt considerate costisitoare. În consecință, aceste teste sînt rareori utilizate ca un mijloc de a detecta o eroare primară. Astfel, nu este practic să se utilizeze un test de diagnostic al unei memorii cu acces aleator după fiecare acces. În schimb, un astfel de test poate fi rulat periodic, de exemplu la pornirea unui sistem sau pentru rezervele în așteptare. Această strategie este folosită în unele sisteme tolerante la defectări, cum ar fi sistemul PLURIBUS[27], la care testele de diagnostic *off-line* sînt rulate pentru unitățile redondante în așteptare ale sistemului.

Testele prin multiplicare cer în general multiplicarea activității sistemului testat. Tipul de multiplicare utilizat poate lua diferite forme, ce depind de tipul defectării care a fost anticipată. Cea mai comună utilizare a testelor realizate prin multiplicare constă în detecția defectelor componente lor fizice ale sistemului hardware. Dacă se consideră că nu există erori de proiectare și că defectările componentelor hardware apar independent, atunci multiplicarea poate lua forma unei copii separate, dar identice cu a sistemului original, care funcționează în paralel cu aceasta, obținîndu-se două mulțimi de semnale de ieșire ce pot fi comparate de un test simplu de egalitate; rezultă în acest fel o structură autotestabilă prin dublare. Este clar, că acest mod de a realiza testarea nu permite identificarea defectărilor rezultate din proiectare, care ar afecta atît originalul cît și copia. Se poate interveni și asupra acestor tipuri de defectări dîndu-se, atunci cînd este posibil, versiuni diferite originalului și copiei.

Testele realizate prin inversarea funcțiilor sistemului sînt utilizate în acele sisteme pentru care relațiile dintre intrări și ieșiri sînt simple. Aceste teste se realizează prin sinteza semnalelor de intrare din semnalele de ieșire ale sistemului, care sînt apoi comparate cu semnalele de la intrările sistemului, evidențiindu-se în acest mod prezența sau absența unor defectări ale sistemului; rezultă astfel o structură autotestabilă (§ 3.3). De exemplu, această categorie de teste este aplicată în cazul sistemului ESS 1 A[43] pentru operația de scriere a benzii magnetice; se citesc înapoi datele scrise pe bandă și se compară apoi cu datele originale.

Este clar că testele de inversiune pot fi aplicate sistemelor la care inversarea funcțiilor de ieșire poate fi realizată cu ușurință. Pentru acest tip de sisteme o astfel de testare este ușor de implementat și are avantajul de a fi independentă de proiectarea și implementarea sistemului funcțional.

Dintre *testele* prin *codare*, testele de paritate sînt cele mai cunoscute. Pentru *testele de paritate* un singur bit este asociat unui set de biți astfel că suma modulo doi a biților unui cuvînt este zero (paritate impară) sau 1 (paritate pară). Astfel de teste sînt utilizate în sistemele tolerante la defectări ESS No. 1 A, PLURIBUS, TANDEM 16 pentru detectarea defectărilor din sistemele de memorie și a erorilor din transmisiile între unitățile componente. De asemenea, mai pot fi utilizate codurile redondante de tip Hamming, codurile ciclice [36] sau codurile aritmetice. Acestea din urmă se folosesc de exemplu pentru testarea operațiilor aritmetice ale procesoarelor, a operațiilor de memorare sau în transmisiile de date.

Astfel, sistemul tolerant la defectări JPL-STAR[5] utilizează codurile aritmetice pentru testarea funcțiilor aritmetice, dar folosește o structură autotestabilă prin dublare pentru testarea operațiilor logice. Codurile aritmetice sînt mai puțin utilizate în sistemele tolerante la defectări actuale,

pentru că există tendința unui hardware ieftin, iar un singur cip pentru procesor face imposibilă tratarea separată a operațiilor logice de cele aritmetice.

Comparate cu alte forme de testare, testele prin codare sînt considerate ca o metodă eficientă de detecție a erorilor, mai ales în termenii redondanței cerute pentru implementarea testării.

3.2. METODE DE GENERARE A SECVENȚELOR DE TEST PENTRU DIAGNOSTICAREA DEFECTELOR

3.2.1. DEFINIȚII ȘI NOTAȚII

Pentru dezvoltările ulterioare, se definesc în continuare următoarele concepte:

Intrări și ieșiri primare: conexiuni de intrare și ieșire ale sistemului, accesibile exteriorului.

T e s t e : succesiunea vectorilor-semnale de intrare, $\mathbf{X}_k$, aplicați sistemului la intrările primare și, respectiv, a vectorilor-semnale de ieșire, $\mathbf{Y}_k$ prezenți la ieșirile primare:

$$T_k = \{x_{k1}, x_{k2},\ldots,x_{kn}; y_{k1}, y_{k2},\ldots,y_{km}\},\ k = 1,2,\ldots,s. \qquad (3.2)$$

Secvența de test: succesiunea semnalelor la intrările primare.

Lungimea testului, este dată de numărul s.

Rezoluția de diagnosticare a testului reprezintă cantitatea de informații furnizată de mulțimea testelor de diagnostic.

În general, se deosebesc următoarele trei categorii de teste pentru diagnoză:

- *Test parametric*: test care implică verificarea — în cazul circuitelor electrice de exemplu — a unor caracteristici parametrice, cum ar fi tensiuni, impedanțe etc.
- *Test funcțional*: test care implică verificarea funcțiilor logice ale circuitului — în cazul circuitelor logice, de exemplu. În acest scop se folosesc dicționare de defectări, care permit verificarea conformității funcțiilor-răspuns corespunzătoare.

La rîndul lui testul funcțional poate fi:

— *exhaustiv*, cînd tabelul de adevăr caracteristic sau tabelul tranzițiilor — în cazul circuitelor logice secvențiale — este în totalitate verificat;

— *parțial*, cînd verificarea se efectuează pentru anumite combinații sau secvențe ale semnalelor de intrare, în particular pentru acelea care intervin în funcționarea curentă a circuitului respectiv;

— *statistic*, cînd secvențele de test sînt date aleator cu o lege de repartiție uniformă.

- *Test dinamic*, test care implică și verificarea timpului de răspuns și de comutație în condiții reale de utilizare.

Se subliniază că în cazul circuitelor logice cele mai multe ipoteze care stau la baza metodelor de generare a secvențelor de test privitoare la natura defectărilor* iau în considerație defectările logice și cel mai adesea defectările care au ca efect blocarea uneia dintre ieșiri în „0" sau „1" logic.

---

* Considerîndu-se natura defectărilor se disting: *defectări logice*, defectări care conduc la o modificare a tabelului de adevăr al circuitului și *defectări de propagare* (impulsuri parazite sau tranziții eronate).

Ipoteza cea mai utilizată în practică pentru generarea secvențelor de test este cea a defectărilor de blocare; se menționează faptul că numeroase alte categorii de defectare pot fi exprimate ca o formă de blocare [20]. De asemenea, majoritatea metodelor de sinteză a testelor se bazează doar pe ipoteza defectărilor unice. Totuși această ipoteză este necorespunzătoare pentru sistemele redondante, pentru care trebuie avută în vedere — în general — posibilitatea apariției defectărilor multiple [38].

3.2.2. METODE DE GENERARE A SECVENȚELOR DE TEST PENTRU CIRCUITELE LOGICE

La ora actuală, există o diversitate de metode de detecție și localizare a defectărilor în circuitele logice. Unele dintre aceste metode nu prezintă decît un interes pur teoretic, în timp ce altele au condus la realizări practice deosebite. Comparația acestor metode este dificilă, pentru că fiecare dintre metode depinde de diversitatea de ipoteze inițiale avute în vedere. În continuare se vor analiza critic cele mai reprezentative metode de testare în scopul evidențierii particularităților lor și posibilităților de utilizare. Prin exemple se va evidenția atît utilitatea metodelor studiate, cît și unele utilizări necorespunzătoare ale diferitelor teste.

Alegerea uneia sau a alteia dintre metode face apel la mai multe criterii, dintre care trebuie avute în vedere mai ales următoarele:

- *natura circuitului*: secvențial sau combinațional (în pofida unor cercetări intense nu există încă o metodă unică aplicabilă ambelor tipuri de circuite);
- *dimensiunea sistemului*: numărul de circuite elementare;
- *tehnologia utilizată*: circuite integrate pe scară redusă, medie, largă sau foarte largă, tehnologie TTL, MOS etc.;
- *informația cercetată*: detecția sau diagnosticarea defectărilor.

Metodele de generare a secvențelor de test utilizabile pentru circuitele logice se pot clasifica în [15], [47]:

- *metode analitice*, respectiv *funcționale*, după cum la generarea secvențelor de test se ține seama de structura circuitelor logice, respectiv numai de funcția lor logică;
- *metode deterministe*, respectiv *probabilistice*, după cum generarea lor se bazează pe principii deterministe, respectiv probabilistice.

În continuare metodele de generare a secvențelor de test vor fi clasificate după cel de-al doilea criteriu, precizîndu-se totodată dacă sînt metode analitice sau funcționale.

3.2.2.1. GENERAREA SECVENȚELOR DE TEST PRIN METODE DETERMINISTE

Metodele deterministe de generare a secvențelor de test pot rezulta prin analiza și simularea funcțiilor logice ale circuitului, respectiv prin sinteza funcțiilor logice ale circuitului.

A. Generarea secvențelor de test prin analiza și simularea funcțiilor logice

Combinația semnalelor de test la intrările primare este impusă pe baza analizei structurii circuitului, iar prin simularea comportării acestuia în prezența anumitor defectări se determină defectările susceptibile de a fi puse în evidență cu secvența de test respectivă la intrările primare.

O astfel de metodă necesită un program optim de simulare a defectărilor circuitului analizat, deoarece altfel numărul iterațiilor ar fi ridicat, ceea ce ar face ca timpul de lucru să devină, în cazul rulării unui test exhaustiv, prohibitiv.

1°. *Simulare deductivă.* După o partiționare a circuitului în circuite elementare și funcțional disjuncte se dezvoltă o combinație de teste pentru fiecare dintre circuitele elementare pe baza logicii circuitului, iar apoi se stabilesc semnalele de detecție la intrările primare ale circuitului. Metoda este eficace pentru circuitele simple, dar nu este realistă pentru circuitele mai complexe. În ultimul caz, secvențele de test generate în acest mod ar putea fi incomplete sau foarte lungi, cu o redondanță inutilă și cu posibile erori.

Exemplul 3.1. *Metoda căii sensibile**. Ideea metodei constă [4] în alegerea cîtorva căi de propagare de la un punct de manifestare a defectului spre ieșiri. Fie acest punct poarta G_0, iar calea fiind formată din porțile $G_1 \ldots G_n$ (fig. 3.2), astfel stabilite încît semnalele la ieșirea oricărei porți G_j să fie determinate numai de către semnalul de intrare care vine de la poarta G_{j-1}. În felul acesta semnalul la ieșirea porții G_0 va fi dedus doar din observarea semnalului de la ieșirea porții G_n. Calea de porti $G_0, G_1, \ldots, G_n$ astfel stabilită va fi denumită *cale sensibilă* [4], [38 .

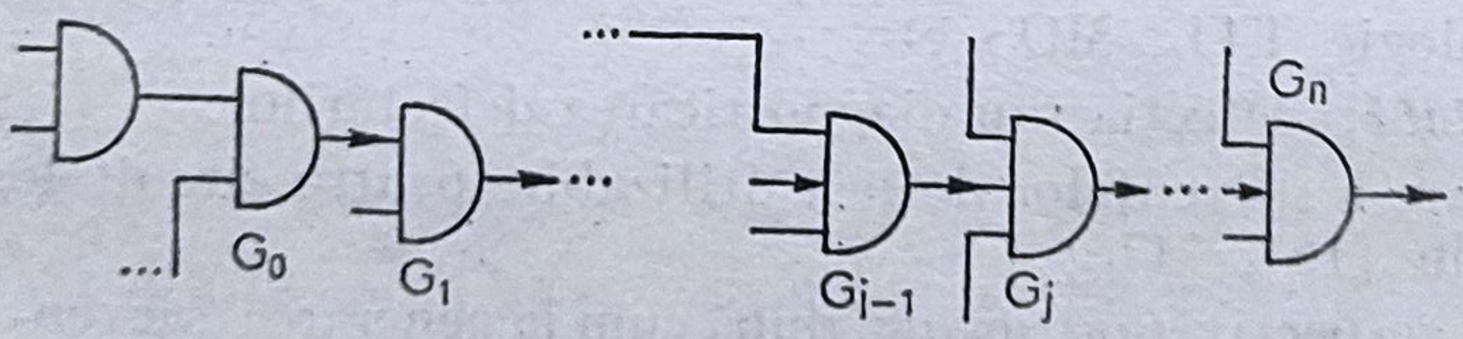


Fig. 3.2. Cale sensibilă de propagare a unei defectări spre ieșirea circuitului.

În final, se deduc semnalele de intrare în poarta G_0 funcție de semnalele impuse la ieșirea porții G_0.

Problema testării în cazul acestei metode poate fi formulată astfel: avînd un set de „căi sensibile" trebuie găsită o combinație de semnale la intrările primare care va realiza toate intrările necesare acestei căi de porți.

În continuare, se va evidenția aplicabilitatea metodei în cazul circuitelor logice.

Exemplul 3.2. Se consideră circuitul din figura 3.3 cu defectarea: ieșirea G_7 blocată în „1". Trebuie stabilit un test care să evidențieze defectul în termenii corespunzători intrării și ieșirii porții G_7.

---

* Cale sensibilă la propagarea unei defect.

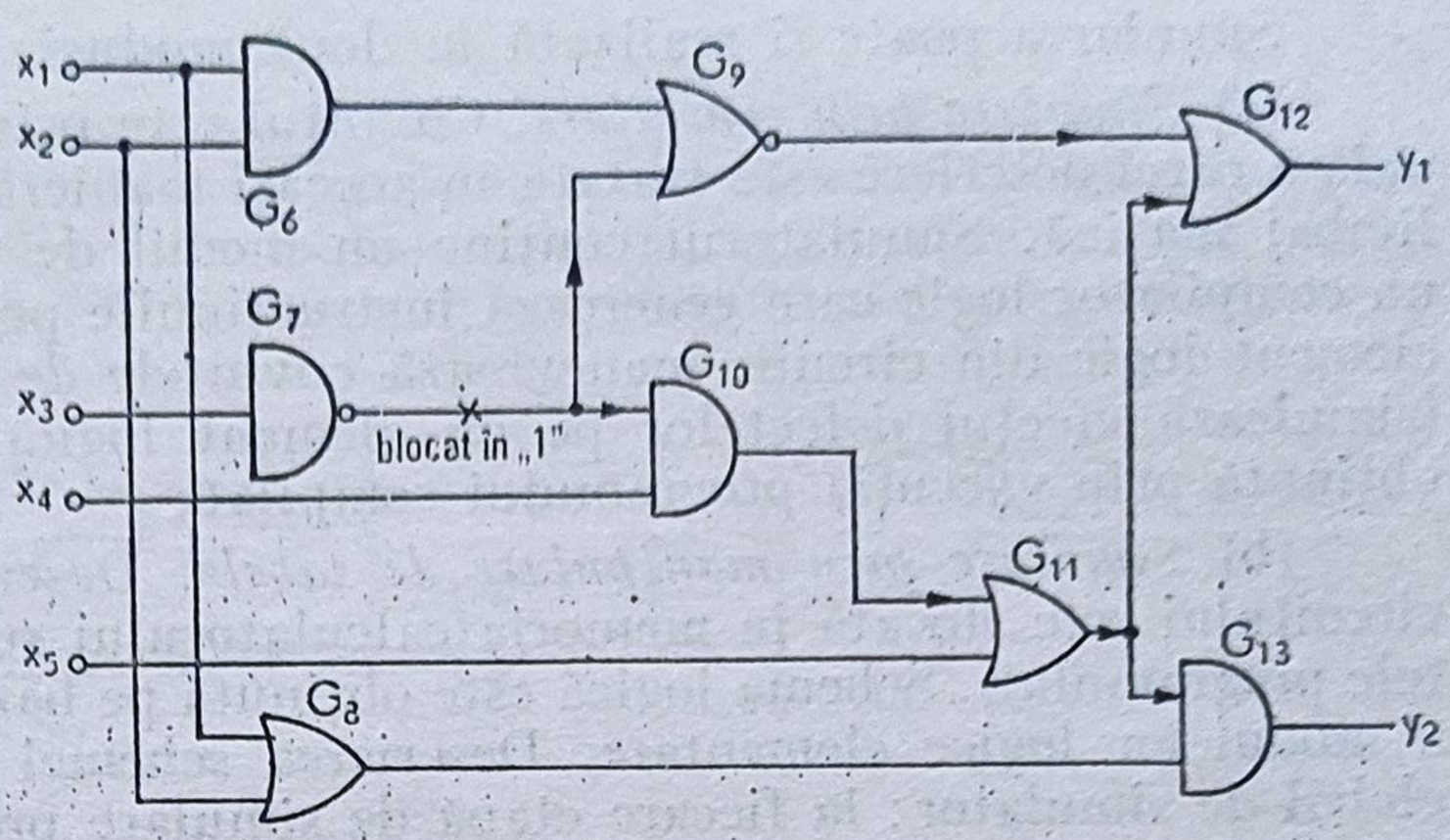


Fig. 3.3. Aplicarea metodei „căii sensibile" pentru identificarea unui test de diagnoză.

Astfel, semnalul $x_3 = 1$ va pune în evidență defectarea porții G_7. În acest caz $y_{G_7} = 0$, dacă defectul este absent și, respectiv, $y_{G_7} = 1$, dacă defectul este prezent.

Într-o primă etapă se alege arbitrar calea G_7, G_{10}, G_{11}, G_{13}, de la G_7 la ieșire. Condițiile necesare pentru ca aceasta să fie o cale sensibilă de propagare a defectării porții G_7 spre ieșirea circuitului sînt următoarele: $x_4 = 1$, pentru a sensibiliza poarta G_{10}, $x_5 = 0$, pentru a sensibiliza poarta G_{11} și $y_{G_8} = 1$, pentru a sensibiliza poarta G_{13}.

În etapa a doua a procedeului se găsește vectorul test în termenii intrărilor primare, după cum urmează:

Din prima etapă s-au stabilit următoarele semnale la intrările primare ale circuitului: $x_3 = 1$; $x_4 = 1$; $x_5 = 0$.

Pentru a avea semnalul $y_{G_8} = 1$, trebuie ca $x_1 = 1$ și $x_2 = 0$.

S-a obținut astfel vectorul test:

$T_1 = (x_1, x_2, x_3, x_4, x_5; y_2) = (1, 0, 1, 1, 0; 1)$, care identifică defectarea: ieșirea porții G_7 blocată în „1". Atunci cînd semnalul $y_2 = 1$ se va evidenția prezența defectării, iar cînd $y_2 = 0$, absența defectării.

Defectarea nu se observă la ieșirea y_1, deoarece căile G_7, G_9, G_{12} și G_7, G_{10}, G_{11}, G_{12} nu sînt sensibile la propagarea defectării considerate. Astfel, $y_{G_9} = 0$ și $y_{G_{11}} = 1$ implică $y_1 = 1$, atunci cînd defectul este prezent și $y_{G_9} = 1$ și $y_{G_{11}} = 1$, atunci cînd defectul nu este prezent, implică aceeași valoare, $y_1 = 1$.

Această situație se întîlnește atunci cînd căile de propagare diverg de la locul defectului și converg la ieșire, numărul de inversiuni pe căile respective fiind neegal.

De menționat că metoda denumită a *căii sensibile* nu conduce întotdeauna, respectiv pentru orice tip de circuit logic, la un test de diagnostic.

2°. *Analiză și simulare asistată de calculator*. În acest scop se pornește de la un model matematic al simulării funcționării circuitului (sistemului), dezvoltat pe calculator, introducîndu-se succesiv, într-un mod sistematic, diferite defectări prin modelul lor — pentru care se dezvoltă testele corespunzătoare [25], [8].

Simularea poate fi realizată în două moduri:

(a) *Simulare prin compilare*. Circuitul este privit ca o unitate funcțională a cărei descriere este tratată în aceeași manieră ca un program scris în limbaj mașină. Simulatorul conține un modul de prelucrare preliminară, un compilator logic care generează instrucțiunile printru simularea fiecărui element logic din circuit, manevrează cuvintele de injectare a defectărilor (simulează efectul defectelor pe un element logic). Simularea este astfel obținută prin execuția programului compilat;

(b) *Simulare prin manipulare de tabele*. Descrierea schemei logice a circuitului este stocată în memoria calculatorului sub formă de tabele (datele programului). Schema logică este obținută pe baza descrierii funcționale a modulelor logice elementare. Descrierea schemei logice nu este tratată global de simulator; la fiecare etapă de simulare programul determină elementul care trebuie tratat, fiind apelat subprogramul potrivit.

Acest mod de lucru prezintă o serie de avantaje. Astfel, există posibilitatea folosirii unor trasee selective de porți în circuit, la un moment dat fiind considerate numai modulele pentru care una dintre intrări și-a modificat nivelul logic; de aceea, procedeul are ca rezultat o reducere importantă a timpului de lucru. De asemenea, el prezintă o mare suplețe ca mod de lucru; introducerea unui nou modul este simplă, fiind suficientă modificarea instrucțiunii de găsire a funcțiilor logice și introducerea subprogramului corespunzător simulării noului tip de modul.

Metoda prezintă o mare generalitate, este aplicabilă circuitelor mari și poate detecta un mare număr de defectări.

B. Metode de generare a secvențelor de test prin sinteză

În acest caz sînt determinate, pe baza considerării structurii interne a circuitului cu ajutorul unui algoritm — pentru un defect — , una sau mai multe combinații de semnale de detecție.

În cele ce urmează se va exemplifica aplicarea algoritmului *D*[41] ca metodă de generare a secvențelor de test prin sinteză.

1°. *Metoda algoritmului D*. Algoritmul *D* [41] conduce la obținerea unui test pentru diagnosticarea unei defectări în termenii intrării și ieșirii porții defecte, generînd simultan toate căile posibile de propagare a defectării la toate ieșirile primare ale circuitului.

La fiecare pas al algoritmului se verifică convergența căilor, renunțîndu-se la dezvoltarea în continuare a căii care nu este divergentă (operație numită *propagare*). În final se urmăresc pînă la intrările primare toate căile de propagare generate, căutîndu-se în mod automat valorile logice ale semnalelor de intrare, care evidențiază manifestarea defectărilor la ieșirile primare.

E x e m p l u l 3. 3. În continuare se vor evidenția etapele algoritmului, considerîndu-se circuitul din figura 3.4, pentru care nu s-a putut obține un test de diagnoză, utilizîndu-se o metodă de analiză [7].

Algoritmul utilizat presupune următoarele etape:

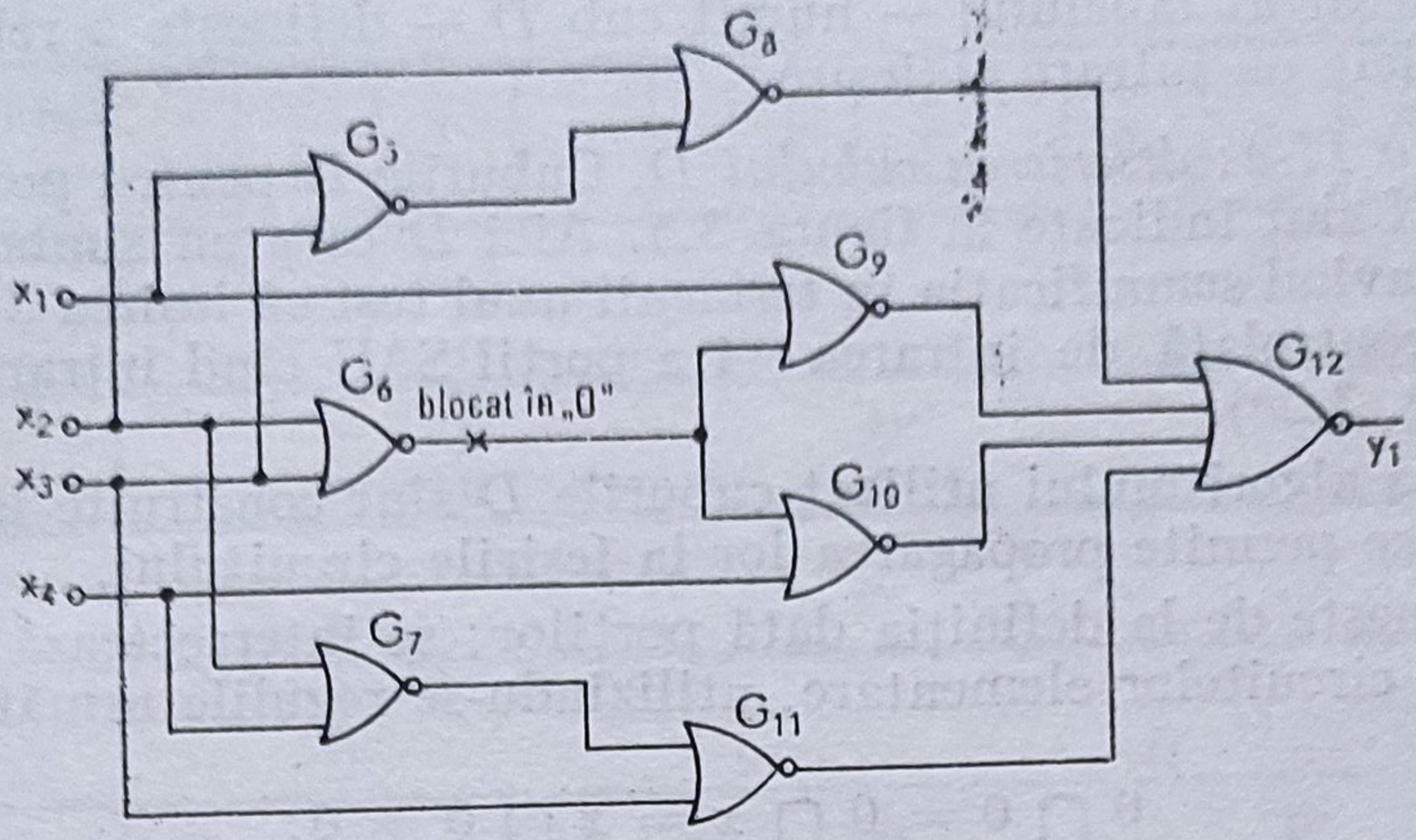


Fig. 3.4. Circuit pentru exemplificarea algoritmului *D*.

Etapa I: *descrierea porților logice cu tabelul lor de adevăr*. Pentru circuitul analizat porțile *G* sînt definite în tabelul 3.1. În acest tabel, pe lîngă utilizarea valorilor logice 0 și 1 s-a mai introdus simbolul *x*, avînd semnificația că linia corespunzătoare din circuit poate lua orice valoare, 0 sau 1.

Tabelul 3.1

Poarta \ Nr. nod	1	2	3	4	5	6	7	8	9	10	11	12
G_5	*x*		1		0							
	1		*x*		0							
	0		0		1							
G_6	*x*	1			0							
	1	*x*			0							
	0	0			1							
G_7	*x*		1			0						
	1		*x*			0						
	0		0			1						
G_8	*x*			1			0					
	1			*x*			0					
	0			0			1					
G_9	1				*x*			0				
	x				1			0				
	0				0			1				
G_{10}		1			*x*				0			
		x			1				0			
		0			0				1			
G_{11}	*x*				1					0		
	1				*x*					0		
	0				0					1		
G_{12}					*x*	*x*	*x*			1	*x*	0
					x	*x*	1			*x*	*x*	0
					x	1	*x*			*x*	*x*	0
					1	*x*	*x*			*x*	*x*	0
					0	0	0			0	0	1

Fiecare şir al tabelului — numit cub D — defineşte o relaţie cauzală între semnalele de intrare şi ieşire.

Etapa a II-a: *descrierea cubului D*. Cuburile D pentru porţile elementare SAU, ŞI sînt indicate în figura 3.5. Aici D este un simbol care poate fi 0 sau 1, avînd semnificaţia în termenii unui test că ieşirea 3 — de exemplu — este controlată de intrarea 1 a porţii SAU cînd intrarea 2 are valoarea 0 (fig. 3.5*a*).

Pe baza algoritmului utilizat cuburile D sînt construite în mod sistematic, ceea ce permite propagarea lor la ieşirile circuitului.

Se porneşte de la definiţia dată porţilor; se intersectează cuburile de definiţie ale circuitelor elementare, utilizîndu-se regulile următoare:

$$\begin{aligned} &0 \cap 0 = 0 \cap x = x \cap 0 = 0; \\ &1 \cap 1 = 1 \cap x = x \cap 1 = 1; \\ &x \cap x = x; \\ &1 \cap 0 = D; \\ &0 \cap 1 = D'. \end{aligned} \tag{3.3}$$

Pentru circuitul considerat, cuburile D — construite cu relaţiile (3.3) aplicate cuburiior de definiţie indicate în tabelul 3.1 — sînt indicate în tabelul 3.2.

Cuburile obţinute evidenţiază uşor semnalele intrărilor, care pot controla ieşirea fiecărei porţi.

Etapa a III-a: *Construirea cubului defectării*. Construirea acestui cub se bazează pe acelaşi formalism ca la faza anterioară. Cubul unei defectări permite exprimarea testelor în vederea diagnosticării unui defect în termenii intrării şi ieşirii porţii defecte.

Pentru circuitul tratat cubul defectării — ieşirea porţii G_6 blocată în „0" — este indicat în tabelul 3.3, avînd semnificaţia că atunci cînd intrările 2 şi 3 sînt fixate în O, ieşirea 6 va avea valoarea 0 dacă defectarea este prezentă şi 1 dacă defectarea este absentă.

Etapa a IV-a: *Generarea căii sensibile*. Generarea unei căi de propagare pentru cubul defectării — cale sensibilă — este obţinută cu ajutorul intersecţiei D [40], definită în tabelul 3.4.

O b s e r v a ţ i e. Simbolurile Φ şi ψ semnifică faptul că intersecţia D nu are sens, respectiv este nedefinită ■

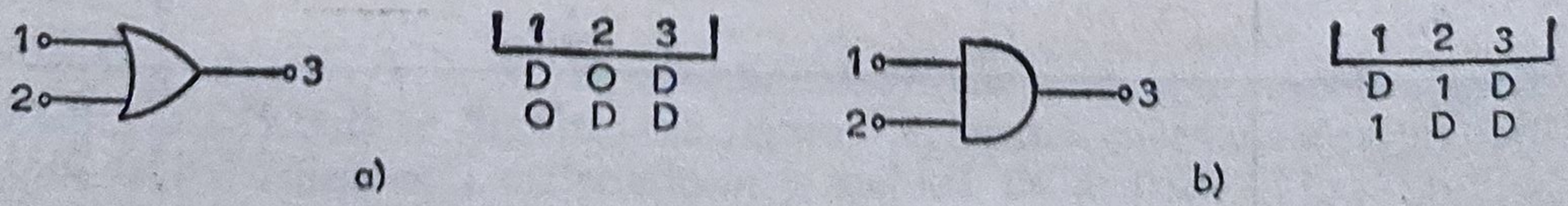


Fig. 3.5. Cuburile D pentru:
a — poarta logică SAU; *b* — poarta logică ŞI.

Tabelul 3.2

Poarta \ Nr. nod	1	2	3	4	5	6	7	8	9	10	11	12
G_5	0 D		D 0		D' D'							
G_6		0 D	D 0			D' D'						
G_7		0 D		D 0			D' D'					
G_8		0 D			D 0			D' D'				
G_9	0 D					D 0		D' D'				
G_{10}			0 D	D 0				D' D'				
G_{11}			0 D		D 0				D' D'			
G_{12}						D 0 0 0	0 D 0 0	0 0 D 0	0 0 0 D	D' D' D' D'		

Tabelul 3.3

Cubul defectării												Eticheta	Poarta	Ieşirea diverge la	Comentarii
1	2	3	4	5	6	7	8	9	10	11	12				
0	0				D							CS°	G_6	9,10	Se trece la etapa a IV-a

Tabelul 3.4

$\cap$ D	0	1	x	D	D'
0	0	Φ	0	ψ	ψ
1	Φ	1	1	ψ	D'
x	0	1	x	D	η
D	ψ	ψ	D	η	ζ
D'	ψ	ψ	D'	ζ	η

În cazul circuitului analizat generarea căii sensibile este efectuată utilizîndu-se intersecţia D a cubului defectării cu cuburile D de propagare ale porţilor G_9 şi G_{10}.

Intersecţia D a expresiei $CS°$ (tabelul 3.3) cu al doilea cub D de propagare (tabelul 3.2) al porţii G_9, conduce la rezultatul:

$$\frac{|1\ 2\ 3\ 6\ 9|}{x\ 0\ 0\ D\ x} \cap_D \frac{|1\ 2\ 3\ 6\ 9|}{0\ x\ x\ D\ D'} = \frac{|1\ 2\ 3\ 6\ 9|}{0\ 0\ 0\ \eta\ D'} \qquad (3.4)$$

Semnificația rezultatului obținut anterior este următoarea: semnalul de ieșire va fi complementul logic al semnalului apărut în punctul 6 al circuitului, deoarece dacă într-o intersecție D, apare numai η dar nu și ζ, atunci vor exista:

$$D \underset{D}{\cap} D = D \qquad \text{și} \qquad D' \underset{D}{\cap} D' = D'. \tag{3.5}$$

Prin urmare,

$$\frac{|\ 1\ 2\ 3\ 6\ 9\ |}{x\ 0\ 0\ D\ x} \underset{D}{\cap} \frac{|\ 1\ 2\ 3\ 6\ \ 9|}{0\ x\ x\ D\ D'} = \frac{|\ 1\ 2\ 3\ 6\ \ 9|}{0\ 0\ 0\ D\ D'} \tag{3.6}$$

S-a obținut astfel propagarea cubului D al defectului prin poarta G_9 a circuitului. Propagarea cubului D al defectului prin poarta G_{10} se obține în mod similar. Se continuă operația de generare a căilor sensibile prin poarta G_{12}. Se aplică intersecția D căii sensibile $CS^{0,1}$ și se obține:

$$CS^{0,1} \underset{D}{\cap} \{12\} = \frac{|\ 1\ 2\ 3\ 6\ 8\ 9\ 10\ 11\ 12\ |}{0\ 0\ 0\ D\ x\ D'\ x\ \ x\ \ x} \underset{D}{\cap} \frac{|\ 1\ 2\ 3\ 6\ 8\ 9\ 10\ 11\ 12\ |}{x\ x\ x\ x\ 0\ D\ 0\ 0\ D'} =$$

$$= \frac{|\ 1\ 2\ 3\ 6\ 8\ 9\ 10\ 11\ 12\ |}{0\ 0\ 0\ D\ 0\ \zeta\ 0\ \ 0\ D'} \tag{3.7}$$

Simbolul ξ are următoarea semnificație: ieșirea porții G_{12} este controlată de nodul 9 din circuit, astfel încît semnalul de ieșire la poarta G_{12} este complementul semnalului din punctul *9* al circuitului. Rezultatele sînt concentrate în tabelul 3.5.

Termenul $CS^{0,1,2}$ nu este dezvoltat mai departe cu acest algoritm, întrucît, pentru această cale de propagare nu există o ieșire divergentă (din acest motiv în tabelul 3.5 apare simbolul Φ).

Etapa a V-a: Determinarea testului. Secvența de test se determină pornindu-se de la fiecare cale sensibilă obținută în etapa anterioară spre intrările primare. Se scriu semnalele logice corespunzătoare după cum urmează:

Se începe cu calea $CS^{0,1,2}$, obținută mai sus. Se selectează elementul de valoare 0 sau 1 cel mai apropiat de ieșirea circuitului (este cazul ieșirii porții G_{11} a circuitului). Se determină semnalele la intrarea porții G_{11}, astfel încît să se obțină la ieșirea acestei porți semnalul „0". Se continuă în acest mod, din aproape în aproape, pînă la intrările primare, cînd se concluzionează că nu există un test corespunzător pentru această cale. Rezultatele sînt concentrate în tabelul 3.6.

Procedîndu-se în mod similar și pentru $CS^{0,2,2}$ se obține același rezultat.

În schimb, pentru $CS^{0,1,1,1}$ s-a obținut testul T:

T : (0, 0, 0, 0, 0).

Pe baza algoritmului prezentat a putut fi obținut un test de diagnoză efectuîndu-se — grație cubului D — propagarea în paralel pe toate căile posibile a efectului defectării considerate.

Metoda cubului D poate fi aplicată, cu unele modificări [40], [41], nu numai circuitelor logice combinaționale, ci și circuitelor logice secvențiale.

2°. *Metoda diferențelor booleene*. Metoda se bazează pe principiul sensitivizării căilor din circuit și conduce la o procedură de testare uniflux.

Diferența booleană este definită ca fiind rezultatul aplicării operatorului SAU EXCLUSIV funcției logice a sistemului, reprezentînd comporta-

Tabelul 3.5

Nodul	Intersecția D (1 2 3 4 5 6 7 8 9 10 11 12)	Eticheta	Poarta logică	Ieșirea diverge la	Comentarii
CS^0	0 0 D	CS^0	6	9, 10	Cubul D al defectării din circuit
$CS^0 \underset{D}{\cap} \{9\}$	0 0 0 D D'	$CS^{0,1}$	{6,9}	9, 10, 12	Prima cale sensibilă prin G_9
$CS^0 \underset{D}{\cap} \{10\}$	0 0 0 D D'	$CS^{0,2}$	{6,10}	9,10,12	A doua cale sensibilă prin G_{10}
$CS^{0,1} \underset{D}{\cap} \{10\}$	0 0 0 D D' D'	$CS^{0,1,1}$	{9,10}	12	Cale sensibilă prin G_9 și G_{10}
$CS^{0,1} \underset{D}{\cap} \{12\}$	0 0 0 D 0 D' 0 0 D	$CS^{0,1\,2}$	{12}	—	Se trece la determinarea testului
$CS^{0,2} \underset{D}{\cap} \{9\}$	0 0 0 0 D D' D'	$CS^{0,2,1}$	{9,10}	12	Identic cu $CS^{0,1,1}$
$CS^{0,2} \underset{D}{\cap} \{12\}$	0 0 0 D 0 0 D' 0 D	$CS^{0,2,2}$	{12}	—	Se trece la determinarea testului
$CS^{0,1,1} \underset{D}{\cap} \{12\}$	0 0 0 0 D 0 D' D' 0 D	$CS^{0,1,1,1}$	{12}	—	Nu se mai caută o cale de propagare, se trece la căutarea testului

mentul bun, și funcției logice a sistemului testat în prezența unui defect. Dacă funcția rezultată nu este nulă înseamnă că s-a obținut o secvență de test.

Metoda permite obținerea secvențelor de test pentru orice tip de defectare, dar este eficace doar pentru circuitele de mică/medie complexitate [28], [38].

Tabelul 3.6

Nodul	Intersecția D (1 2 3 4 5 6 7 8 9 10 11 12)	Comentarii
1	2	3
$CS^{0,1,2}$	0 0 0 D 0 D' 0 0 D	Ieșirea 11 trebuie să fie „0". Cu tabelul 3.1 se identifică două posibilități: $G_{11}(a)$ și $G_{11}(b)$. $G_{11}(b)$ este în contradicție cu $CS^{0,1,2}$; se alege $G_{11}(a)$
$G_{11}(a)$ =	1	
$G_{11}(b)$ =	x 0	

Tabelul 3.6 (continuare)

1	2	3
$CS^{0,1,2} \cap_D G_{11}(a) =$ $G_{10}(a) =$ $G_{10}(b) =$	0 0 0 D 1 0 D' 0 0 D 1 x 0 x 1 0	Ieşirea *10* trebuie să fie „0". Cu tabelul 3.1 se identifică două posibilităţi: $G_{10}(a)$ şi $G_{10}(b)$. $G_{10}(b)$ este în contradicţie cu $CS^{0,1,2} \cap_D G_{11}(a)$; se alege $G_{10}(a)$.
$CS^{0,1,2} \cap_D G_{11}(a) \cap_D G_{10}(a) =$ $G_8(a) =$ $G_8(b) =$	0 0 0 1 D 1 0 D' 0 0 D 1 x 0 x 1 0	Ieşirea *8* trebuie să fie „0". Cu tabelul 3.1 se identifică două posibilităţi: $G_8(a)$ şi $G_8(b)$. $G_8(a)$ este în contradicţie cu $CS^{0,1,2} \cap_D G_{11}(a) \cap_D G_{10}(a)$; se alege $G_8(b)$.
$CS^{0,1,2} \cap_D G_{11}(a) \cap_D G_{10}(a) \cap_D G_8(b) =$ $G_7(a) =$	0 0 0 1 1 D 1 0 D' 0 0 D 0 0 1	Ieşirea *7* trebuie să fie „1". Există numai o posibilitate şi ea este în contradicţie cu $CS^{0,1,2} \cap_D G_{11}(a) \cap_D G_{10}(a) \cap_D G_8(b)$. Nu există un test corespunzător pentru $CS^{0,1,2}$.
$CS^{0,1,1,1} =$ $G_{11}(a) =$ $G_{11}(b) =$	0 0 0 0 D 0 D' D' 0 D x 1 0 1 x 0	Start cu $CS^{0,1,1,1}$. Ieşirea *11* trebuie să fie „0". Cu tabelul 3.1 se identifică două posibilităţi: $G_{11}(a)$ şi $G_{11}(b)$. $G_{11}(b)$ este în contradicţie cu $CS^{0,1,1,1}$; se alege $G_{11}(a)$.
$CS^{0,1,1,1} \cap_D G_{11}(a) =$ $G_8(a) =$ $G_8(b) =$	0 0 0 0 D 1 0 D' D' 0 D 1 x 0 x 1 0	Ieşirea *8* trebuie să fie „0". Există două posibilităţi: $G_8(a)$ şi $G_8(b)$. $G_8(a)$ este în contradicţie cu $CS^{0,1,1,1} \cap_D G_{11}(a)$; se alege $G_8(b)$.
$CS^{0,1,1,1} \cap_D G_{11}(a) \cap_D G_8(b) =$ $G_7(a) =$	0 0 0 0 1 D 1 0 D' D' 0 D 0 0 0 1	Ieşirea *7* trebuie să fie „1". Există o soluţie.
$CS^{0,1,1,1} \cap_D G_{11}(a) \cap_D G_8(b) \cap_D G_7(a) =$ $G_5(a) =$	0 0 0 0 1 D 1 0 D' D' 0 D 0 0 0 1	Ieşirea *5* trebuie să fie 1. Există o soluţie.
Testul valid	0 0 0 0 1 D 1 0 D' D' 0 D	Au fost verificate porţile căii $CS^{0,1,1,1}$ şi a rezultat un test.

3.2.2.2. GENERAREA SECVENȚELOR DE TEST PE PRINCIPII PROBABILISTICE

Se evidențiază două metode: metoda de generare aleatoare a testelor și, respectiv, metoda de simulare de tip Monte Carlo.

Prin natura nedeterministă a vectorilor de test trebuie înțeles că aceștia nu sînt generați pe baza unui algoritm, prin care se urmărește evidențierea unor defectări anume. Detectarea erorilor se bazează pe natura aleatoare și numărul mare al vectorilor de test.

A. Metoda generării aleatoare a testelor

Intrările circuitului testat și cele ale unui circuit martor sînt baleiate în paralel în momentul efectiv al testării de către secvențele de test generate aleator. Semnalele de ieșire ale celor două circuite sînt comparate: în caz de divergență a celor doi vectori ai semnalelor de ieșire, circuitul (sistemul) testat este considerat defect. Schema de aplicare a metodei este indicată în figura 3.6.

Secvențele de detecție obținute permit stabilirea unui *dicționar al defectărilor* pentru circuitul (sistemul) respectiv [24]; de aceea procedeul poate fi considerat de tip autoinstruibil.

O problemă care se pune în legătură cu această metodă este aceea a determinării lungimii unei secvențe de test, L, care asigură detecția defectărilor prezente cu o probabilitate dată [24]. Evident, lungimea secvenței de test, L, depinde de structura circuitului testat.

În continuare se notează cu P_A probabilitatea de a accepta după testare un circuit defect și cu P_{BD} probabilitatea de bună funcționare a unui circuit în prezența unei defectări la o anumită combinație a semnalelor de intrare. Lungimea testului, L, trebuie să fie astfel încît după aplicarea a L secvențe de test la intrarea circuitului să existe relația:

$$(P_{BD})^L < P_A\,, \tag{3.8}$$

de unde rezultă:

$$L > \log P_A / \log P_{BD}. \tag{3.9}$$

Valoarea probabilității P_{BD} este dependentă de structura internă și de starea logică a circuitului testat.

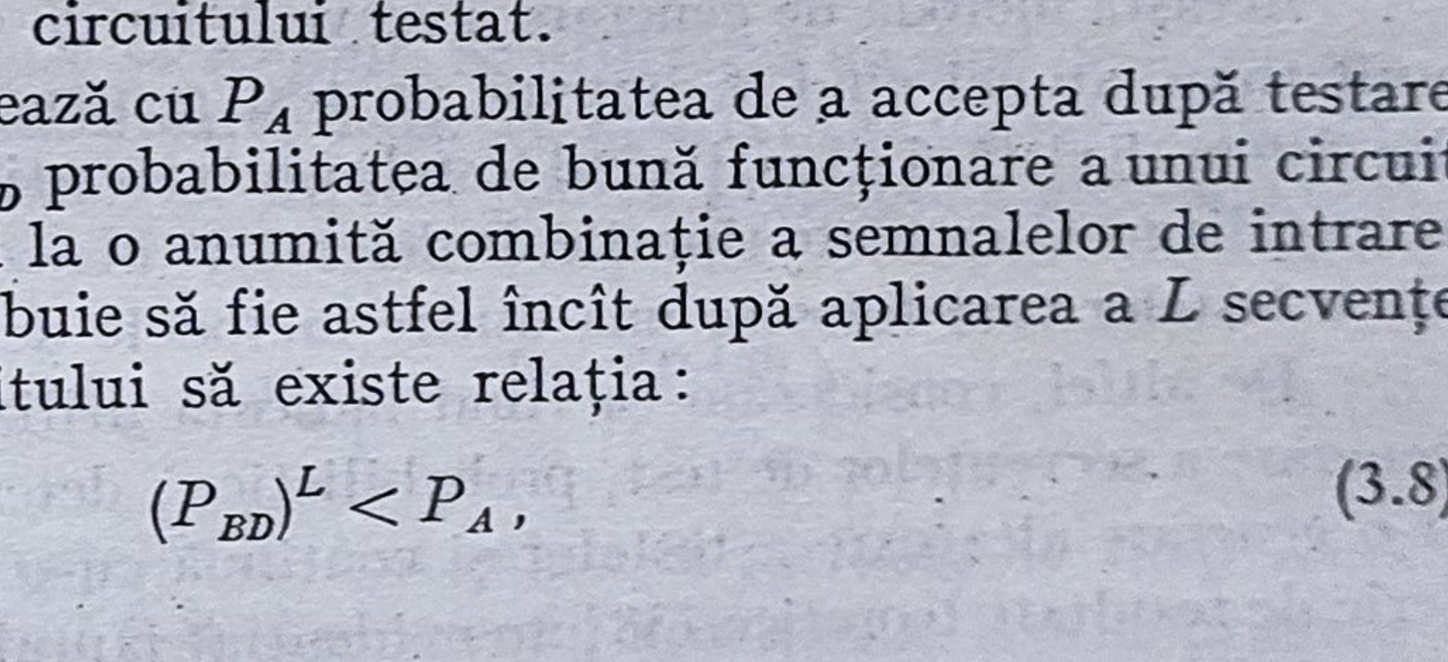


Fig. 3.6. Aplicarea metodei generării aleatoare a testelor.

Metoda este utilă în cazul testării circuitelor integrate pe scară largă, cînd localizarea unui defect în circuit nu mai constituie un obiectiv major, astfel încît este justificată folosirea unei metode de testare de tip „da sau nu", testele fiind generate pe principii probabilistice. De exemplu, aplicîndu-se o metodă a testelor aleatoare la patru arii de porți de complexități diferite (între 550 și 1 000 de porți) se reliefează [46] o creștere asimptotică a numărului defectelor identificate odată cu mărirea numărului vectorilor de test aplicați.

B. **Metoda simulării Monte Carlo**

În esență, metoda constă într-o analiză cu ajutorul unei simulări a defectărilor de detectat. Semnele de intrare sînt obținute aleator — cu o subrutină de generare aleatoare a numerelor 0 și 1. Se determină apoi, cu ajutorul unui simulator de circuite logice cu injecție de defectări, abilitatea fiecărei vector test — combinație a semnalelor de intrare — de a detecta diferitele defectări din circuit, conform unei liste de defectări.

Fie *NDA* numărul de circuite deduse prin introducerea inițială a celor *NDA* defectări presupuse pentru circuitul analizat. Circuitele considerate sînt excitate cu combinații aleatoare de semnale. Cele *NDA* circuite simulate și circuitul martor sînt comparate, pe rînd, pentru fiecare combinație de semnale. La fiecare neconcordanță a semnalelor de ieșire se evidențiază combinația semnalelor-test necesară diagnosticării defectării considerate.

Această metodă de generare a secvențelor de test poate consuma foarte mult timp, atunci cînd se impune un test pentru toate defectările posibile din sistem. Există și situații cînd testele generate utilizîndu-se această metodă nu pot diagnostica toate tipurile de defectări; în acest caz se recomandă generarea testelor care folosesc una dintre metodele deterministe.

De altfel, considerăm ca fiind indicată utilizarea metodelor mixte de generare a secvențelor de test, probabilistice / deterministe, care încep printr-o generare aleatoare a testelor și continuă cu o generare deterministă. O serie de rezultate teoretice [38] au evidențiat faptul că în cazul folosirii doar a metodelor probabilistice de generare a secvențelor de test se obțin teste care la începutul rulării detectează un număr important de defectări, dar pentru care viteza de detecție scade în timp.

Calitatea testelor se poate măsura și prin lungimea secvenței de test. În continuare, se va stabili lungimea unei secvențe de test în cazul metodei de simulare de tip Monte Carlo.

Fiecare combinație de semnale la intrările primare este caracterizată de un factor de merit, P_a, definit ca procentajul defectărilor din circuit pe care testul poate să-l evidențieze. Introducerea acestei noțiuni permite obținerea unei valori a numărului de teste necesare pentru detectarea defectărilor unui circuit cu un anumit nivel de încredere. Se face ipoteza simplificatoare că factorul de merit P_a este constant pe tot timpul rulării testului.

• *Cazul circuitelor logice combinaționale.* Fie $NDA(x)$ numărul de defectări rămase în circuit după aplicarea a x secvențe de test. O nouă secvență de test va detecta $NDA(x)$. P_d defectări, astfel încît numărul defectărilor nedetectate după aplicarea ultimei secvențe de test este:

$$NDA(x)(1 - P_d).$$

Se poate scrie numărul defectărilor nedetectate după aplicarea a L secvențe de test:

$$NDA(L) = (1 - P_d)^L \, NDA(0) \tag{3.10}$$

Dacă se pune condiția conform căreia cele L secvențe de test să detecteze toate defectările din circuit, se poate scrie:

$$NDA(L - 1) \leqslant 1, \tag{3.11}$$

de unde va rezulta lungimea L a testului:

$$NDA(L-1) = (1-P_d)^{L-1} \, NDA(0),$$

$$(1-P_d)^{L-1} \, NDA(0) \leqslant 1,$$

$$(L-1) \log(1-P_d) + \log NDA(0) \leqslant 0,$$

obținîndu-se în final:

$$L \geqslant 1 - \frac{\log NDA(0)}{\log (1-P_d)}. \tag{3.12}$$

• *Cazul circuitelor logice secvențiale.* Se utilizează un mod de lucru analog. Se face corecția corespunzătoare, deoarece în acest caz nu este suficientă o singură combinație a semnalelor de test pentru propagarea efectului unei defectări la ieșirile primare ale circuitului [24]:

$$L \geqslant 1 + L_\Delta + K_1 \left(1 - \frac{\log NDA(0)}{\log (1 - P_d)}\right), \tag{3.13}$$

unde L_Δ este numărul secvențelor de test rulate în perioada de timp scurs între momentul aplicării unui test și momentul apariției la ieșirile primare a confirmării testului, iar K_1 măsoară lungimea medie a fiecărei secvențe parțiale.

3.2.3. STRATEGII DE ELABORARE A SECVENȚELOR DE TEST PENTRU SISTEMELE MARI

Datorită volumului materialului logic care trebuie analizat în scopul stabilirii legăturilor existente între funcțiile realizate de sistem și ansamblul elementelor materiale care îl implementează testarea la nivelul sistemelor mari este complicată. Metodele de generare a secvențelor de test indicate pentru circuitele logice nu pot fi utilizate în acest caz într-un mod competitiv din punctul de vedere al eficacității, costului, rapidității etc.

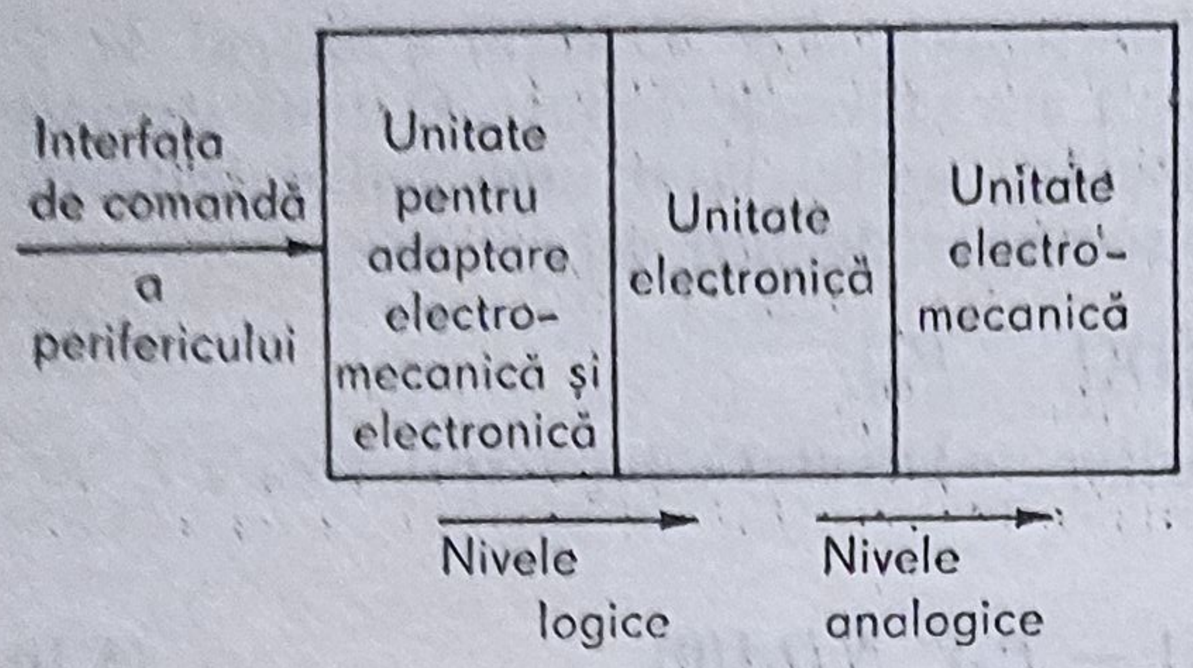


Fig. 3.7. Structura eterogenă a unui sistem complex.

În fig. 3.7 este indicată schema structurii unui sistem complex. Întrucît acest ansamblu are o structură eterogenă, nu este posibilă aplicarea unui *procedeu algoritmic* de generare a vectorilor-semnale de test, asemănător celor utilizate în cazul circuitelor logice, fiind posibilă — din acest motiv — doar aplicarea unor teste de tip *funcțional*.

Astfel, dacă se analizează — de exemplu — echipamentele periferice ale sistemelor de calcul se remarcă că ele nu sînt funcțional independente, ci sînt activate într-o exploatare standard prin intermediul unei unități de legătură de către unitatea centrală a sistemului de calcul în cauză. În consecință, se impune ca verificarea funcțiilor echipamentelor periferice să fie efectuată în condiții de activare similare. Astfel, testarea în vederea detecției și diagnosticării defectărilor se poate face prin derularea de către unitatea centrală a unor programe de test specializate [33].

Avînd în vedere structura eterogenă a sistemelor mari, o altă soluție este partiționarea acestora în module pentru generarea secvențelor de test. În aceste condiții pentru generarea secvențelor de test apar două etape: într-o primă etapă se vor genera secvențele de test la nivelul fiecărui modul, urmînd ca ulterior să se realizeze compunerea acestor secvențe pentru nivelul ansamblului general.

Este necesar ca generarea testelor la nivelul fiecărui modul să ia în considerare compunerea acestor module în structura sistemului, deoarece nu este întotdeauna posibilă propagarea spre ieșire a oricărei secvențe de semnale care conțin o informație în vederea diagnosticării unui defect [29]. Rezultă că *eficacitatea* testului este determinată de modul alegerii modulelor componente ale sistemului considerat.

Se pot distinge două modalități de realizare a acestei partiții a sistemului:

(a) un decupaj *funcțional* al sistemului, facilitînd compunerea modulelor în sistem; în acest caz fiecare modul este caracterizat de o mulțime de operații, care pot fi ușor incluse în fazele de execuție a funcțiilor globale ale sistemului;

(b) un decupaj *structural*, care utilizează proprietăți ca simetria, repetitivitatea etc., de natură să faciliteze și să optimizeze cercetările cu privire la generarea testelor, atît la nivelul modulelor, cît și la nivelul compunerii acestora în ansamblul general.

O partiție optimă a sistemului trebuie să țină seama de cele două orientări. Acest lucru nu este întotdeauna posibil, mai ales dacă sistemul a fost conceput fără a ține cont de posibilitatea testării sale facile. În aceste condiții se realizează un compromis în detrimentul exhaustivității testului sau a lungimii sale.

În procedura de generare a secvențelor de test pot fi evidențiate următoarele trei strategii:

1°. *Elaborarea de teste după principiul start-small*. În această abordare se începe cu testarea celui mai mic modul posibil din sistem, fiecare test suplimentar adăugînd apoi un nou modul pentru testare. Diagrama din figura 3.8 ilustrează secvențele desfășurării metodei.

Cu toate că este relativ complexă, avîndu-se în vedere procedura stabilirii succesiunii testelor individuale, aceste strategii prezintă — după opinia noastră — două avantaje principale și anume:

- localizarea defectărilor este rapidă, dacă se pornește de la accepțiunea că o defectare detectată nu se mai poate găsi într-un circuit nou testat, deoarece nu este posibilă trecerea la pasul următor, decît după înlăturarea defectării respective:
- problema defectărilor multiple, atunci cînd partiționarea sistemului este realizată în module suficient de mici, este rezolvată într-o anumită măsură;

Dificultatea care apare în elaborarea unei astfel de strategii de generare a testelor constă în găsirea unui decupaj adecvat și a unei ordonări a circuitelor rezultate în vederea optimizării testării lor.

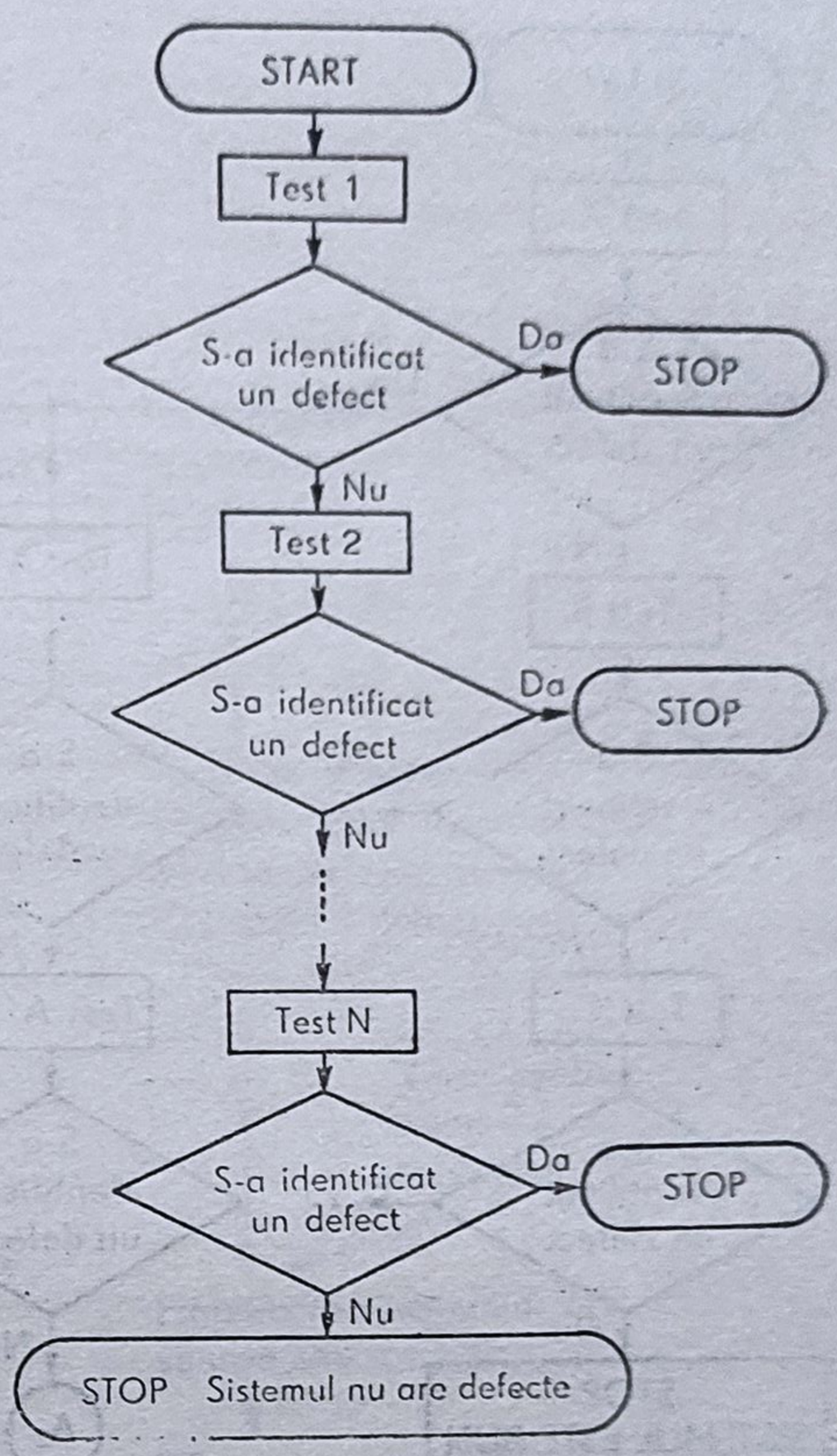


Fig. 3.8. Succesiunea secvențelor algoritmului *start-small*.

2°. *Elaborarea de teste după principiul start-big*. În această situație, testul se aplică — pentru început — unei porțiuni mari din sistem. În caz de bună funcționare a unității testate testul continuă pentru o altă unitate mare, iar dacă se detectează o defectare secvențele de test se pot ramifica pentru a diagnostica decupajele din ce în ce mai fine din sistem pînă la identificarea defectării (fig. 3.9).

Această strategie apare optimală pentru testele efectuate sistematic în vederea unei mentananțe preventive, dar generarea testului este mai complexă decît în prima abordare; mai mult, această abordare nu este aplicabilă în ipoteza defectărilor multiple [18].

3°. *Elaborarea de teste după principiul multiple clue*. Comparativ cu primele două abordări detectarea unei defectări la un moment dat nu întrerupe testarea sistemului și nici nu modifică secvențele de test. Secvențele de test sînt rulate în totalitate, după care se analizează rezultatele în vederea stabilirii unui diagnostic.

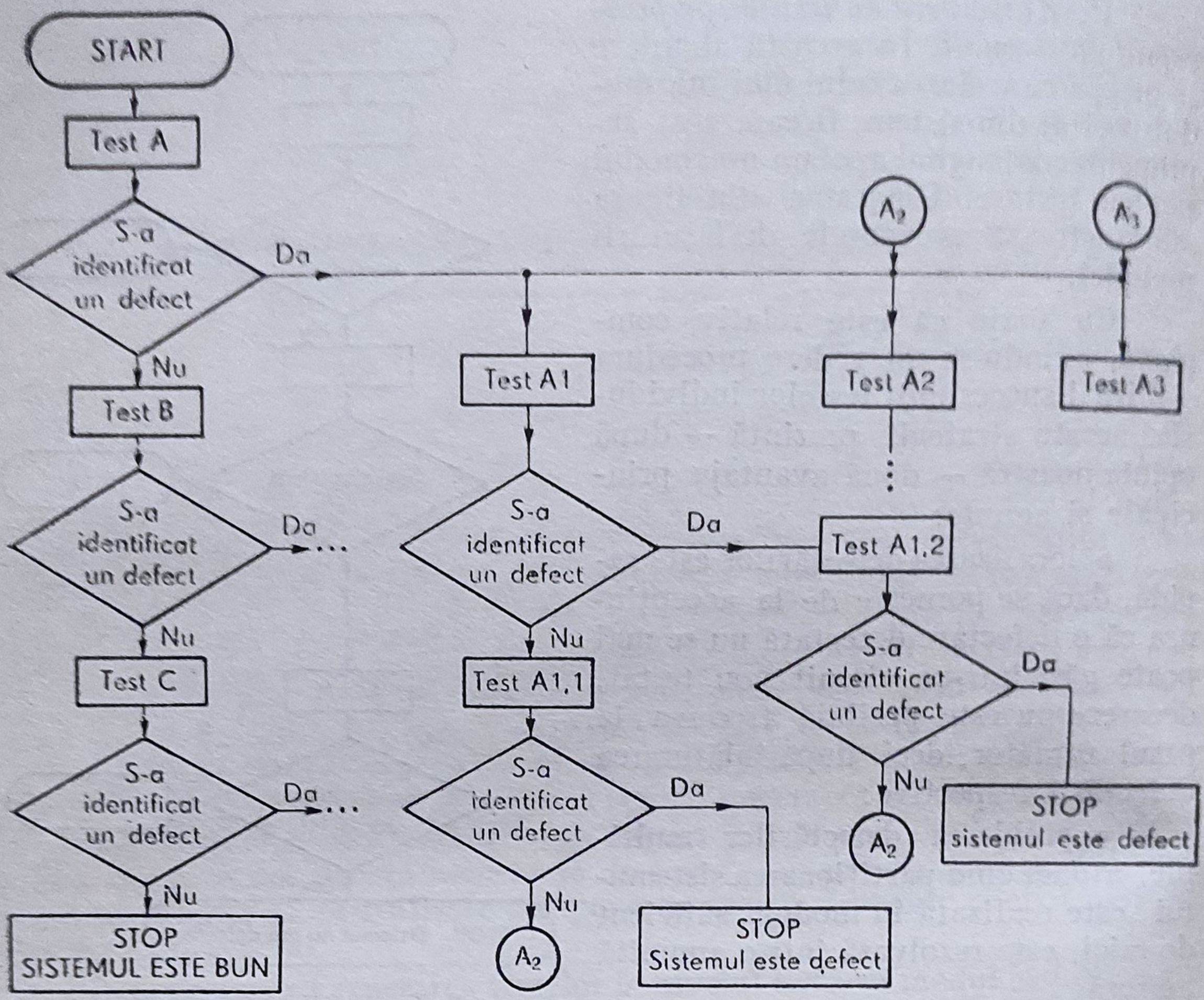


Fig. 3.9. Succesiunea secvențelor algoritmului *start-big*.

Avantajul acestei abordări constă în aceea că nu este necesar un decupaj strict al sistemului în module, nici o ordonare precisă a testelor. În schimb, algoritmii de diagnostic sînt mai sofisticați, iar în ipoteza defectărilor multiple devin mai complicați. În plus, lungimea testului poate fi cîteodată mai mare decît este necesară, căci oricare ar fi momentul detecției unei defectări numărul secvențelor de test este constant.

În acțiunile de mentenanță ale sistemelor, pentru a conferi acestora fiabilitate și flexibilitate, se aplică atît teste de diagnostic cît și teste de verificare a funcționării, precum și teste de încredere. În figura 3.10 este prezentată o metodică [7] propusă pentru aplicarea optimă a testelor generate pentru sistem.

3.2.4. PROCEDEE DE DERULARE A UNUI TEST

La elaborarea unei metode de generare a secvențelor de test pentru un sistem sau chiar pentru un simplu circuit trebuie avută în atenție și metoda de derulare a secvenței de test.

Procedeele de derulare a testelor pentru sistemele digitale se pot clasifica [12] prin analogie cu definițiile date circuitelor logice secvențiale și com-

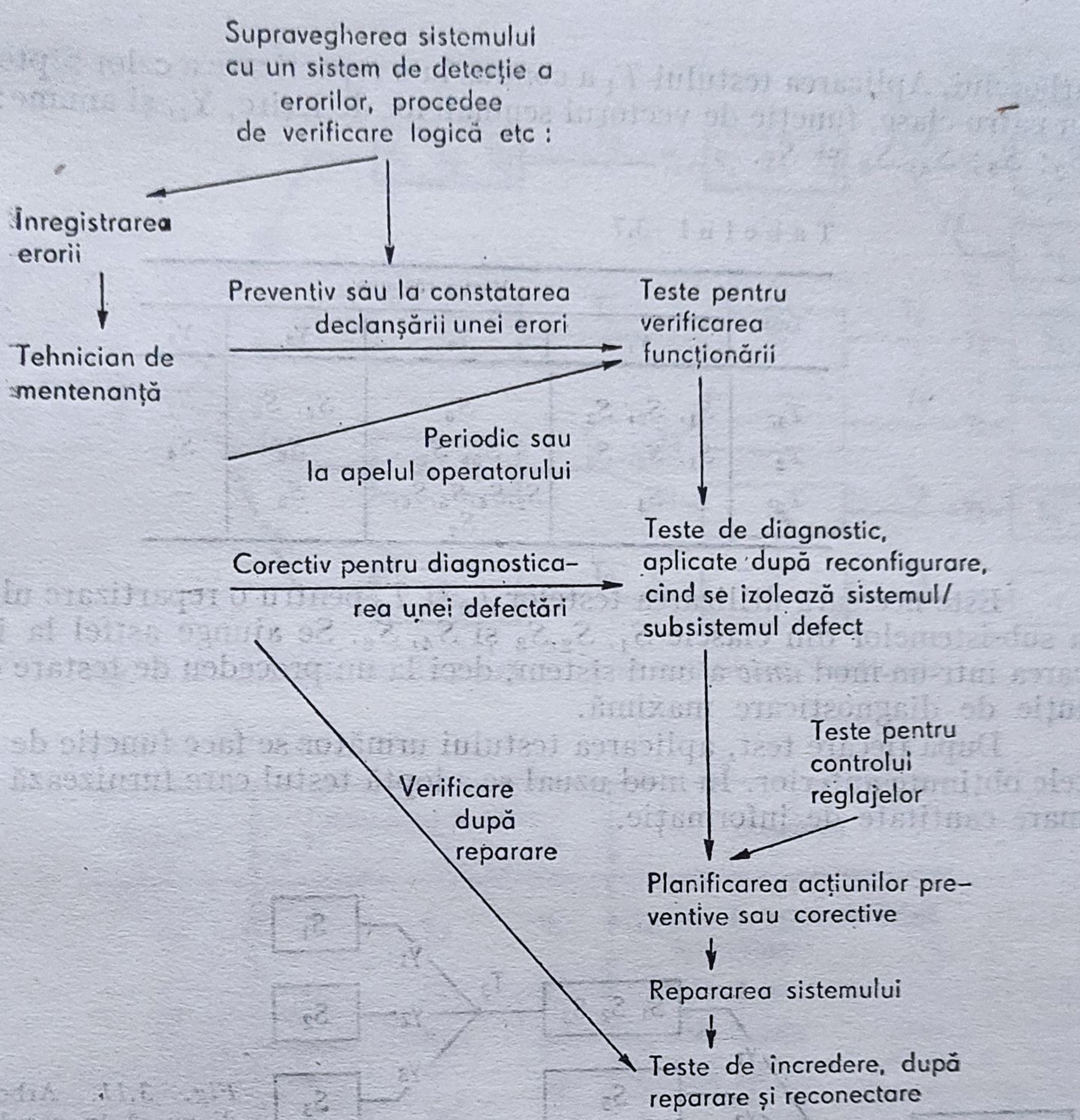


Fig. 3.10. Aplicarea testelor în acțiunile de mentenanță.

binaționale în metode *secvențiale* și, respectiv, *combinaționale*. În cazul testării secvențiale cel de-al i-lea vector test este determinat pe baza răspunsurilor la cei i-1 vectori-test anteriori. În cazul testării combinaționale cel de-al i-lea vector-test este independent de vectorii-test anteriori.

1°. *Metoda testării secvențiale*. În cele ce urmează, principiul metodei este evidențiat pentru un sistem partiționat în șapte subsisteme identice,, S_1 $S_2,\ldots,S_7$, cărora li se aplică trei secvențe de test, T_1, T_2, T_3 (tabelul 3.7), rezultînd patru tipuri de semnale-răspuns, $\mathbf{Y}_1$, $\mathbf{Y}_2$, $\mathbf{Y}_3$, $\mathbf{Y}_4$.

Pe baza datelor din tabelul 3.7 se poate construi diagrama-test din figura 3.11. Se obține un arbore de diagnostic în care fiecare defectare corespunde unei ramuri, dar nu în mod necesar și invers, iar terminațiile ramurilor corespund locurilor unde testarea s-a oprit, cînd s-a identificat o defectare. În cazul analizat, a fost cunoscut apriori că modulul în stare de bună funcționare este S_1 și, prin convenție, acesta a fost plasat pe ramura superioară a

arborelui. Aplicarea testului T_1 a condus la o repartizare a celor şapte module în patru clase, funcţie de vectorul semnalelor de ieşire, Y_i, şi anume: S_1, S_2, S_3; S_5; S_4, S_6 şi S_7.

Tabelul 3.7

Test	Vector-răspuns			
	Y_1	Y_2	Y_3	Y_4
T_1	S_1, S_2, S_3	S_5	S_4, S_6	S_7
T_2	S_1, S_2, S_3	S_5, S_7	S_6	S_4
T_3	S_1	S_2, S_4, S_5, S_6, S_7	S_3	—

Este necesară utilizarea testelor T_2 şi T_3 pentru o repartizare ulterioară a subsistemelor din clasele S_1, S_2, S_3 şi S_4, S_6. Se ajunge astfel la identificarea într-un mod *unic* a unui sistem, deci la un procedeu de testare cu rezoluţie de diagnosticare maximă.

După fiecare test, aplicarea testului următor se face funcţie de rezultatele obţinute anterior. În mod uzual se adoptă testul care furnizează cea mai mare cantitate de informaţie.

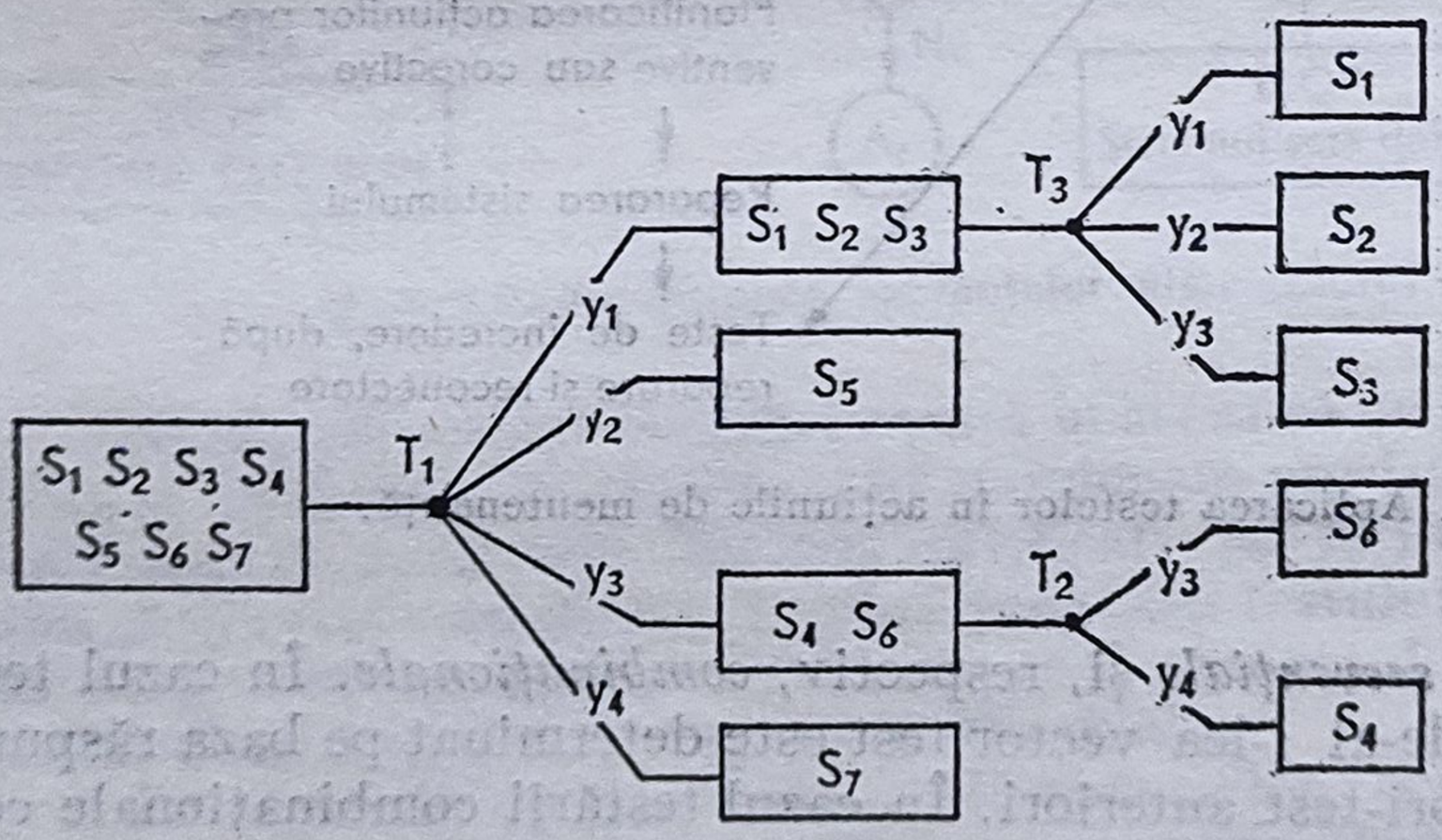


Fig. 3.11. Arborele de diagnoză în cazul testării secvenţiale.

2°. *Metoda testării combinaţionale*. Utilizarea acestei metode are ca rezultat reducerea cantităţii de date stocate in vederea diagnosticării. Creşte însă numărul testelor necesare în raport cu metoda testării secvenţiale şi astfel timpul mediu pentru diagnosticarea unei defectări va fi mai mare. Redondanţa proprie acestei testări poate conduce la detectarea unor defectări care n-au fost considerate printre defectările posibile la generarea testelor, ceea ce compensează timpul mai mare necesar testării.

Aplicarea testelor intr-o ordine prestabilită conduce la un dicţionar al defectărilor, care facilitează diagnosticarea ulterioară a acestora. În continuare se consideră acelaşi exemplu ca în cazul anterior, dar testele sînt derulate după un procedeu combinaţional. Se aplică sistemului trei teste, T_1, T_2, T_3. În figura 3.12 este indicată diagrama-test obţinută ca un arbore de diagnoză. Dicţionarul de defectări corespunzător este prezentat în tabelul 3.8. Deoarece.

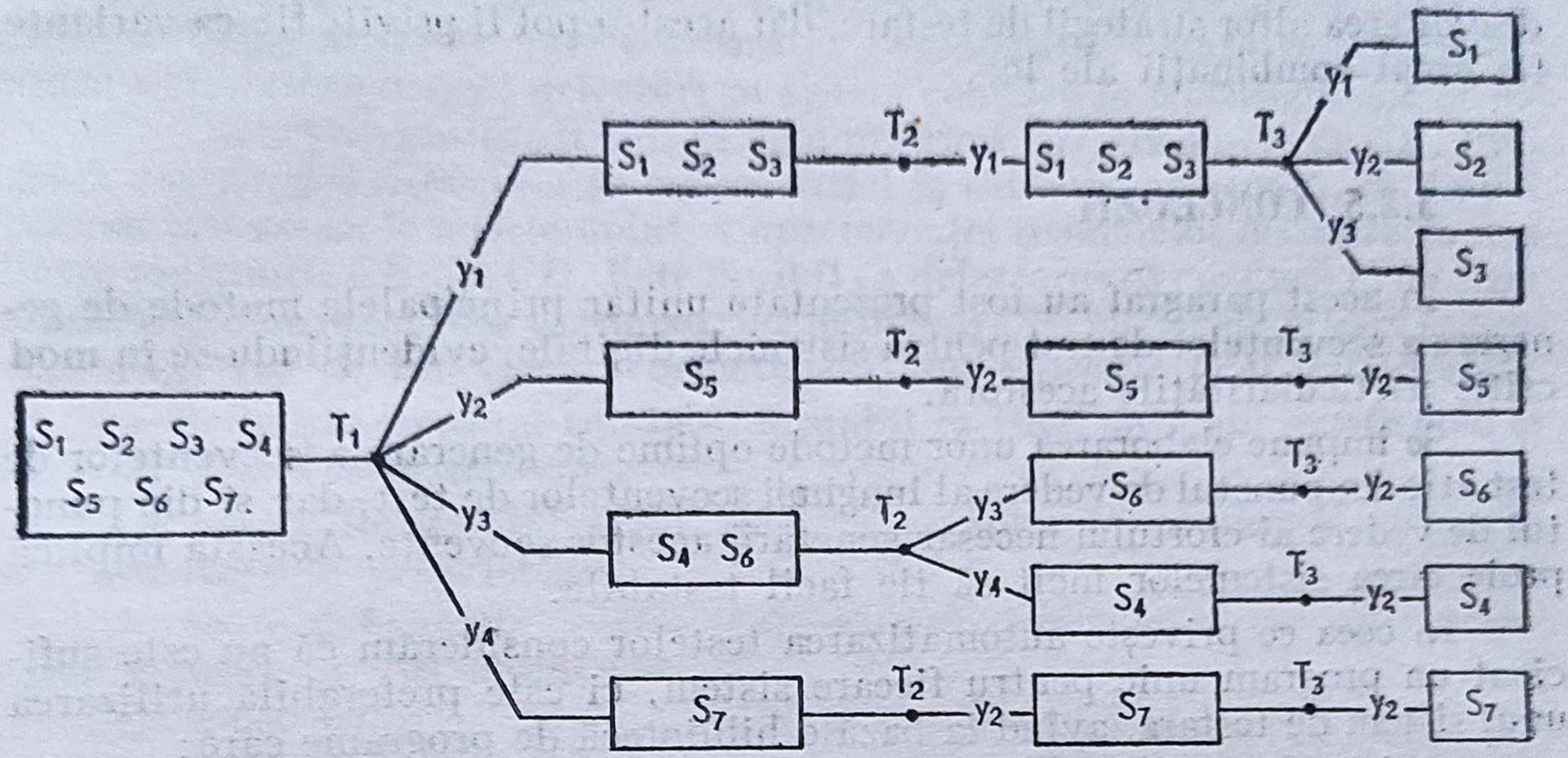


Fig. 3.12. Arborele de diagnoză în cazul testării combinaționale.

Tabelul 3.8

Sisteme	Răspunsuri la testele T_1	T_2	T_3	Comentarii
S_1	Y_1	Y_1	Y_1	S_1 este în stare de bună funcționare
S_2	Y_1	Y_1	Y_2	S_2 este defect
S_3	Y_1	Y_1	Y_3	S_3 este defect
S_4	Y_3	Y_4	Y_2	S_4 este defect
S_5	Y_2	Y_2	Y_2	S_5 este defect
S_6	Y_3	Y_3	Y_3	S_6 este defect
S_7	Y_4	Y_2	Y_2	S_7 este defect

în general, numărul de combinații ale semnalelor care detectează defectările variază de la o defectare la alta, prin ordonarea combinațiilor semnalelor de test în sensul descreșterii numărului de defectări detectante devine posibilă terminarea testării într-un timp mai scurt.

Din cele prezentate anterior rezultă că cele două abordări, secvențială și combinațională, nu sînt echivalente.

Cu toate că prezintă inconvenientul necesității memorării rezultatelor intermediare, metoda testării secvențiale are avantajul de a fi mai rapidă, deoarece sînt aplicate numai combinațiile de test necesare. Dimpotrivă, metoda testării combinaționale nu necesită memorarea rezultatelor testelor, dar este mai lungă, deoarece trebuie reluate toate testele, chiar dacă a fost identificată o defectare.

Alegerea între cele două abordări se face printr-un compromis între mai multe criterii cum sînt gradul de ocupare a memoriei, timpii alocați pentru testare, posibilitatea detecției defectărilor neprevăzute etc.

De menționat că cele două abordări generale descrise pot conduce la dezvoltarea altor strategii de testare, dar acestea pot fi privite fie ca variante, fie drept combinații ale lor.

3.2.5. CONCLUZII

În acest paragraf au fost prezentate unitar principalele metode de generare a secvențelor de test pentru sistemele digitale, evidențiindu-se în mod critic particularitățile acestora.

Se impune elaborarea unor metode optime de generare a secvențelor de test atît din punctul de vedere al lungimii secvențelor de test, dar și din punctul de vedere al efortului necesar generării acestor secvențe. Aceasta implică proiectarea sistemelor încît să fie facil testabile.

În ceea ce privește automatizarea testelor considerăm că nu este suficient un program unic pentru fiecare sistem, ci este preferabilă utilizarea unui sistem de testare, avînd la bază o bibliotecă de programe care:

— folosește proprietățile de simetrie și regularitate a diferitelor circuite ale sistemului;

— facilitează modificările ce trebuie aduse testului ca urmare a unei schimbări de concepție în sistem, etc.

Investigarea prin metode de modelare și simulare presupune descrierea defectelor în termeni de comportament logic bine definit. Cercetările întreprinse în această direcție au permis ca în momentul de față să existe algoritmi — bine puși la punct — de generare automată a vectorilor de test pentru circuite logice combinaționale [41]. Pentru circuitele secvențiale o soluție larg acceptată constă în proiectarea structurată a acestora [8],modalitate care, într-o formă sau alta, reduce problema testării unui circuit secvențial la aceea, bine pusă la punct, a circuitelor combinaționale. Metoda are o mare varietate de implementări, cunoscute sub denumirea generică *Scan Design*.

3.3. SISTEME DIGITALE TOTAL AUTOTESTABILE

3.3.1. DEFINIREA SISTEMELOR AUTOTESTABILE

Fie că este inclus la nivelul unui modul funcțional simplu sau la nivelul întregului sistem, procedeul *autotestării* funcționării sistemului constituie baza implementării tolerării dinamice a defectărilor. Atît sistemele autoreparabile, cît și cele reconfigurabile includ funcția de *autotestabilitate*.

De subliniat că problemele autotestării sau autotestabilității sistemelor digitale sînt diferite de problematica testării sau testabilității ușoare a acestora. Primele presupun testarea sistemelor în funcționare normală. De aceea, secvența de test va fi inclusă în ansamblul configurației sale, sistemul apărînd pentru sarcină că funcționează normal.

Se menționează că implementarea sistemelor tolerante la defectări presupune realizarea unor structuri autotestabile cu o siguranță în funcționare foarte ridicată.

În continuare se consideră că mulțimea semnalelor de ieșire ale sistemului analizat aparține unei mulțimi de configurații, notată în cele ce urmează CE. Apariția unei defectări în sistem conduce la o configurație de ieșire ce nu aparține mulțimii CE, ci unei mulțimi CD, disjuncte cu CE. Problema *autotestării sistemului* se reduce astfel la căutarea automată, în timpul funcționării normale a sistemului, a apartenenței semnalelor de ieșire la una dintre mulțimile CE sau CD. Este posibil ca defectarea să conducă la un semnal de ieșire incorect, dar care aparține mulțimii CE. Sistemele la care se previne acest tip de funcționare defectuoasă sînt *sistemele total autotestabile.*

Se definește sistemul total autotestabil ca fiind sistemul *autotestabil* și *sigur în prezența defectărilor.*

Un sistem total autotestabil relativ la o anumită mulțime de defectări are următoarele proprietăți:

— toate defectările acestei mulțimi sînt detectate în timpul funcționării normale a sistemului;

— orice defectare este detectată de îndată ce a apărut.

Pentru exemplificare se consideră un sistem S cu n intrări care primesc mulțimea semnalelor binare $\mathbf{X} = \{x_i\}$ și m ieșiri funcționale cu mulțimea semnalelor binare $\mathbf{Y}_F = \{y_{Fi}\}$ (fig. 3.13). Necesitatea detecției imediate a defectărilor impune o separare a semnalelor de ieșire în două mulțimi; fie acestea mulțimea semnalelor ieșirilor funcționale, $\mathbf{Y}_F = \{y_{Fi}\}$, în număr de m, și mulțimea semnalelor ieșirilor de test, $\mathbf{Y}_T = \{y_{Tj}\}$, în număr de k.

Configurațiile binare ale variabilelor y_{Tj} corespunzătoare semnalelor de test în funcționare normală aparțin unor configurații de semnale, apriori fixate, notate CE_T.

Pornind de la acest model propus unui sistem în vederea includerii caracteristicii de autotestabilitate, se pot da următoarele definiții referitoare la conceptul de autotestabilitate.

• Un sistem S este autotestabil pentru o mulțime de defectări D dacă și numai dacă pentru orice defectare $d_i \in D$ există un vector al semnalelor de intrare X_i aplicat sistemului S, astfel încît sînt îndeplinite condițiile:

$$\forall\, d_i \in D,\ \mathbf{Y}_T(X_i, d_i) \notin CE_T, \tag{3.14}$$

unde $\mathbf{Y}_T(X_i, d_i)$ este configurația binară a ieșirilor de test rezultată la aplicarea secvenței de intrare X_i sistemului în prezența defectului d_i.

• Există certitudine asupra răspunsului unui sistem autotestabil, conținut în mulțimea semnalelor de test, în prezența unei mulțimi de defecte, D, dacă și numai dacă pentru orice defectare d_i care apare și orice secvență a semnalelor de intrare X_i există relațiile de dependență:

$$\mathbf{Y}(X_i, d_i) = \mathbf{Y}_F(X_i, d_0);\ \mathbf{Y}_T(X_i, d_0) \in CE_T \tag{3.15}$$

sau

$$\mathbf{Y}_F(X_i, d_i) \neq \mathbf{Y}_F(X_i, d_0);\ \mathbf{Y}_T(X_i, d_i) \notin CE_T, \tag{3.16}$$

unde $\mathbf{Y}_F(X_i, d_i)$ este vectorul semnalelor la ieșirile funcționale ale sistemului pentru secvența de intrare X_i aplicată sistemului afectat de o defectare d_i din mulțimea D, iar $\mathbf{Y}_F(X_i, d_0)$, $\mathbf{Y}_T(X_i, d_0)$ sînt vectorii semnalelor de ieșire corespunzători unei funcționări fără defectări.

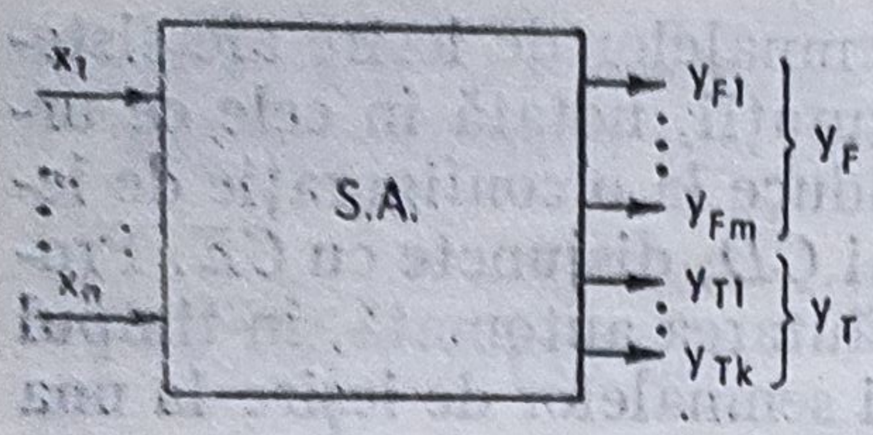


Fig. 3.13. Intrările și ieșirile primare ale unui sistem autotestabil.

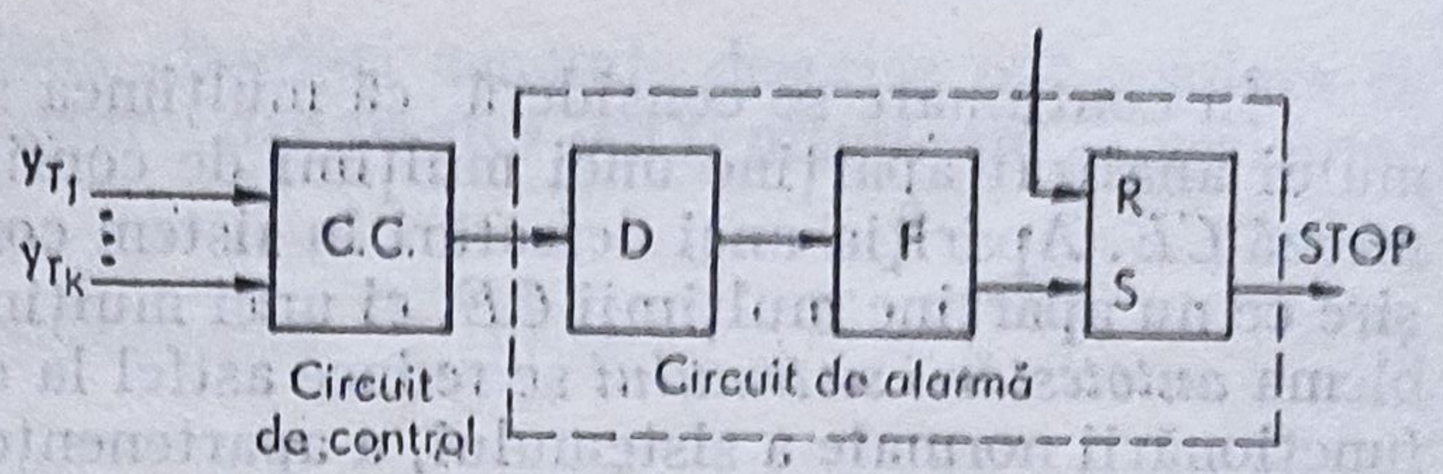


Fig. 3.14. Circuit de detecție a defectării într-un sistem autotestabil.

• Un sistem este total autotestabil pentru o mulțime de defectări D, dacă este autotestabil și există certitudine asupra răspunsului de test în prezența mulțimii de defectări D, adică sînt îndeplinite relațiile (3.14), (3.15) și (3.16).

• *Supravegherea* unui sistem autotestabil poate fi realizată de un sistem de detecție a defectărilor conectat la ieșirile de test. Acesta va forma semnalul de ALARMĂ sau de STOP și va acționa comutarea rezervelor sistemului sau reconfigurarea sa, atunci cînd sistemul are o structură redondantă dinamică, sau va acționa asupra altui sistem de decizie.

Structura unui sistem de detecție care să îndeplinească toate aceste funcții poate fi destul de complexă, mai ales dacă numărul variabilelor y_{Tj} este mare. În figura 3.14 este indicată schema-bloc a unui sistem de detecție, compus — în principal — dintr-un circuit de control și un circuit pentru formarea semnalelor ALARMĂ sau STOP.

Circuitul de control, notat pe schemă cu *C.C.*, trebuie să îndeplinească condițiile:

(a) toate semnalele de test ale sistemului autotestabil controlat $Y_T \notin CE_T$, trebuie să fie observabile la ieșirea sa;

(b) orice defectare care apare în funcționare trebuie să fie detectată în cursul funcționării sale normale.

Aceste condiții sînt îndeplinite dacă:

(1) considerîndu-se că semnalele de intrare ale circuitului de control aparțin mulțimii CE_T, iar semnalele de ieșire aparțin mulțimii CE_c și notîndu-se cu X_{ci} și Y_{ci} vectorii semnalelor de intrare, respectiv de ieșire, ale circuitului de control, există relațiile:

$$\forall X_{ci} \in CE_T, \quad Y_{ci}(X_{ci}) \in CE_c \tag{3.17}$$

și

$$\forall X_{ci} \notin CE_T, \quad Y_{ci}(X_{ci}) \notin CE_c; \tag{3.18}$$

(2) circuitul de control este el însuși de tip autotestabil.

• *Circuitul de formare a semnalului ALARMA sau STOP* trebuie să aibă în structura sa următoarele elemente:

— un circuit care să detecteze configurația semnalelor de la ieșirile circuitului de control care nu aparțin mulțimii CE_c, notat în figura 3.14 cu D;

— un filtru, F, care să elimine eventualele configurații tranzitorii care ar putea să apară în timpul funcționării normale a sistemului;

— un circuit, care asigură înregistrarea erorii detectate de circuitul D, ce poate fi de exemplu un circuit basculant bistabil de tip RS.

Filtrul F este inutil dacă sistemul de detecție este sincronizat de un semnal de tact (cazul sistemelor sincrone).

Trebuie menționat că oricare ar fi realizarea dată unui sistem autotestabil, nu este posibil de detectat toate defectările posibile ale sistemului. De subliniat că realizarea unui sistem autotestabil poate fi condiționată de apariția anumitor tipuri de defectări apriori considerate, funcție de tipul de aplicație avut în vedere pentru sistemul respectiv și tehnologia folosită.

3.3.2. ANALIZA UNOR SISTEME TOTAL AUTOTESTABILE

Realizarea sistemelor total autotestabile are la bază utilizarea structurilor redondante. Aceasta rezultă dintr-o codare explicită sau implicită a informației, ce caracterizează funcționarea sistemului, pentru care s-au utilizat coduri redondante.

În cele ce urmează sistemele total autotestabile vor fi clasificate după tipul redondanței utilizate în: sisteme total autotestabile cu structură redondantă separabilă și, respectiv, sisteme total autotestabile cu structură redondantă neseparabilă.

3.3.2.1. SISTEME TOTAL AUTOTESTABILE CU STRUCTURĂ REDONDANTĂ SEPARABILĂ

Structura generală a unui sistem autotestabil de acest tip este indicată în figura 3.15. Schema analizată are în componență un modul funcțional, notat cu SF, și un modul redondant, notat cu SR. Modulul redondant are ca intrări semnalele $\mathbf{X}$ și $\mathbf{Y}_F$ și formează în funcționare normală semnalele de test $\mathbf{Y}_T$, care aparțin mulțimii $CE_{\bar{T}}$. Deși nu există o soluție generală pentru realizarea unui sistem de tip autotestabil, schema dată în figura 3.15 este utilă, deoarece permite o sistematizare a diferitelor cazuri particulare de sisteme autotestabile.

1°. *Sisteme autotestabile cu structură dublată*. Schema-bloc a unui astfel de sistem este indicată în figura 3.16. Modulul redondant este identic cu cel funcțional. Mulțimea semnalelor de test, $\mathbf{Y}_T$, cuprinde două grupe de semnale corespunzătoare celor două module, funcțional și redondant; aceste semnale sînt comparate în circuitul de detecție a erorilor, iar în caz de necoincidență se formează semnalul de STOP sau ALARMĂ.

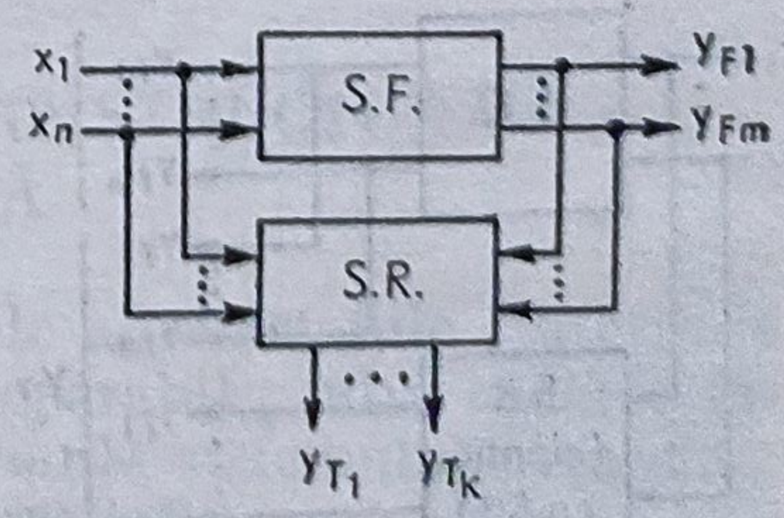


Fig. 3.15. Modelul unui sistem autotestabil cu structură redondantă separabilă.

Structura este foarte simplă și este aplicabilă oricărui tip de sistem. Cu această structură pot fi detectate doar defectări care nu afectează în același timp cele două module ale

sistemului, *SF* și *SR*. Pentru a evita nedetectarea defectelor simultane — cum ar fi cele de concepție — ale celor două module, este recomandabil ca modulul redondant *SR* să fie realizat cu o altă schemă decît modulul funcțional, dar să îndeplinească aceleași funcții.

Deși simplă, această structură de sistem autotestabil nu poate fi acceptată în cele mai multe cazuri, din cauza costului ridicat, anumitor probleme de sincronizare între modulele sistemului, ca și a numărului mare de semnale de test, ceea ce are ca rezultat complicarea circuitelor de control, *C.C.*, corespunzătoare.

2°. *Sisteme autotestabile cu inversarea funcției sistemului de bază*. În acest caz, la ieșirea modulului redondant se obțin semnalele de intrare primare **X** ale sistemului considerat, formate din semnalele de ieșire $\mathbf{Y}_F$ date de modulul funcțional *SF* [23]. Schema-bloc a unui sistem autotestabil astfel conceput este indicată în figura 3.17. Detecția eventualelor defectări este obținută prin compararea informației de intrare în sistem — vectorul semnalelor de intrare — și a informației formate de circuitul de inversare, care constituie aici modulul redondant al sistemului.

De menționat că nu toate sistemele admit un circuit pentru modulul redondant care să realizeze funcția inversă a modulului funcțional. Se pot realiza sisteme autotestabile cu o astfel de structură pentru codificatoare, decodificatoare, dar nu și pentru sumatoare — de exemplu — deoarece în cazul acestora informația primară nu poate fi separată din informația de ieșire.

Concepția acestor sisteme autotestabile este mai complicată decît cea prezentată anterior, modulul redondant având o schemă logică diferită de cea a modulului funcțional. În acest caz, costul sistemului autotestabil depinde de costul sistemului de bază și poate fi superior dublului costului acestuia, iar viteza de funcționare a sistemului este bine limitată de timpii de propagare a informației prin cele două module, funcțional și redondant, și evident va fi mai mică decît în cazul structurii indicate anterior.

3°. *Sisteme autotestabile cu coduri detectoare de erori*. În acest caz modulul redondant al sistemului autotestabil considerat realizează o codare a semnalelor de intrare și ieșire ale modulului funcțional. De exemplu [45],

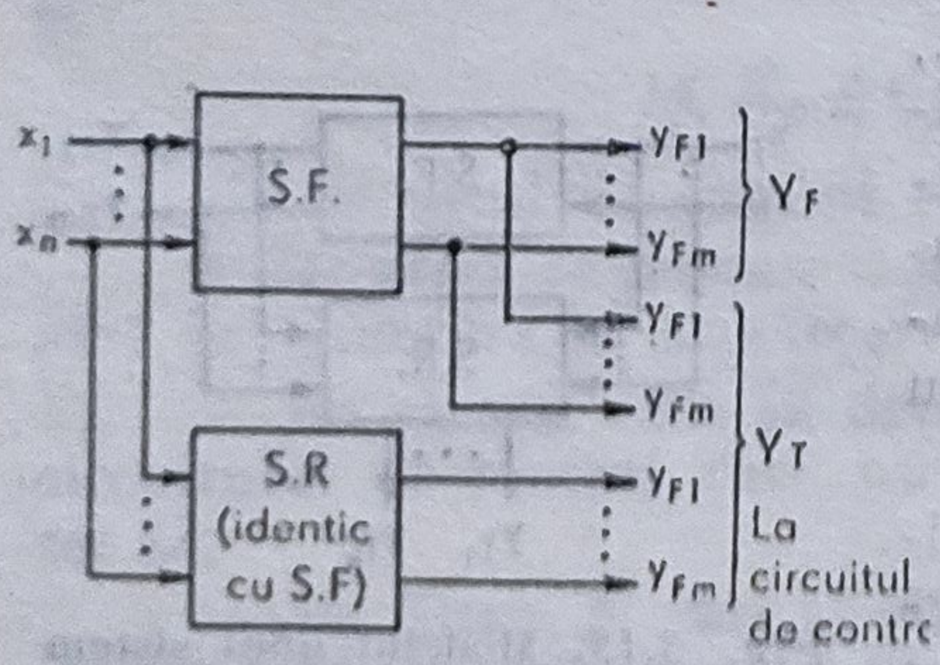


Fig. 3.16. Sistem autotestabil cu structură dublată.

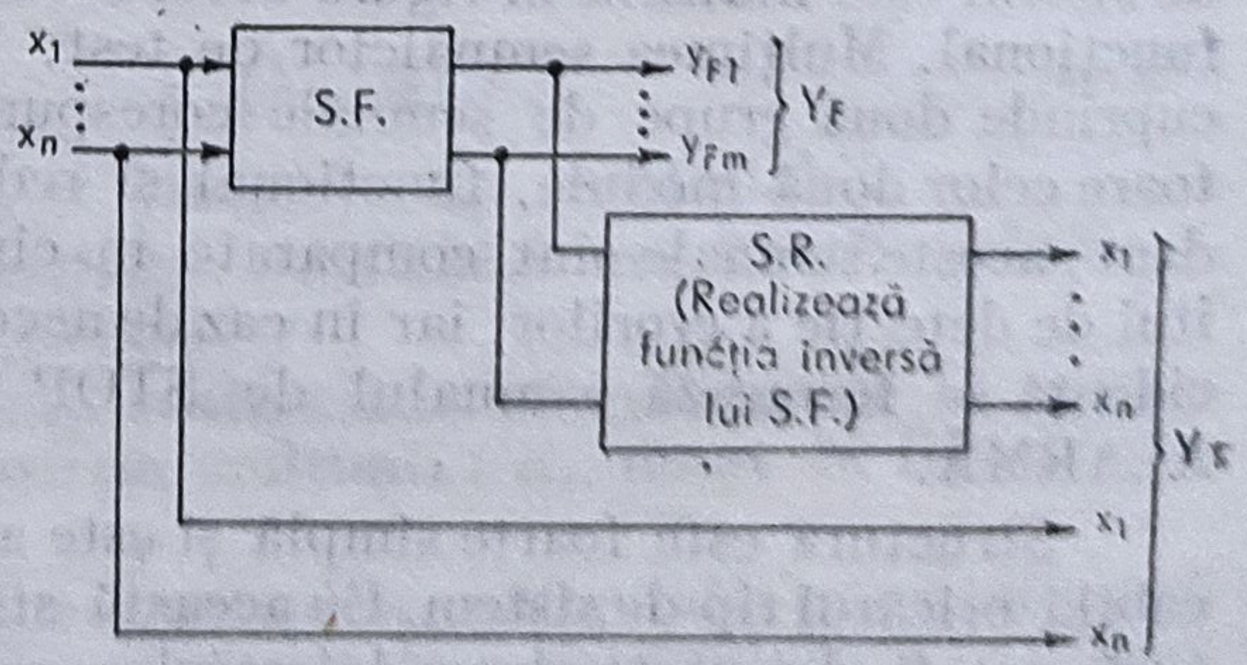
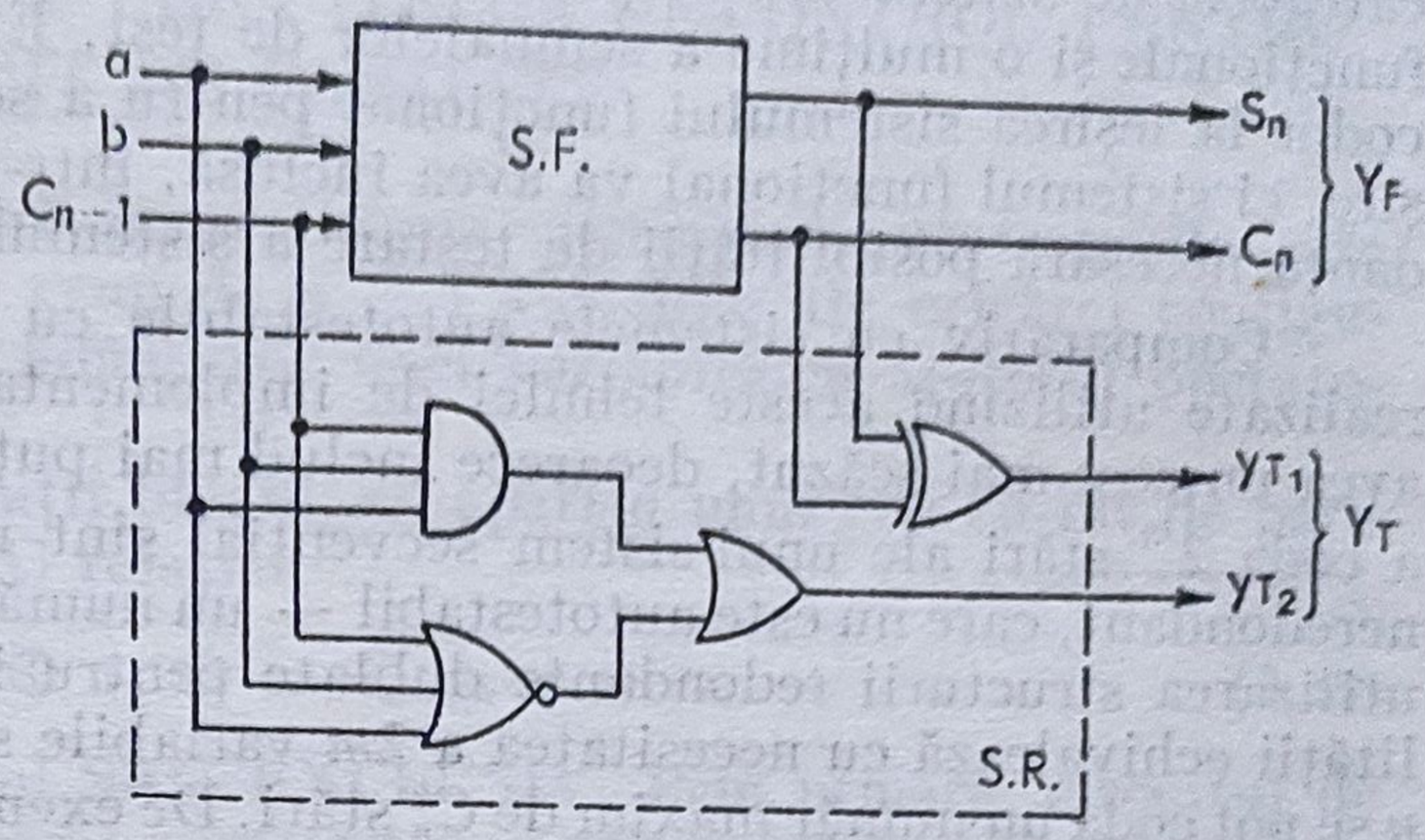


Fig. 3.17. Sistem autotestabil cu inversarea funcției sistemului de bază.

Fig. 3.18. Sistem autotestabil cu coduri detectoare de erori.



pentru sumatorul din figura 3.18 configurațiile a, b, C_{n-1}, S_n, C_n au fost codificate cu două variabile, y_{T_1} și y_{T2}, astfel încît:

$$\{y_{T_1},\ y_{T2}\} = \{0,1\} \text{ sau } \{1,0\}$$

în funcționare normală și

$$\{y_{T_1},\ y_{T_2}\} = \{0,0\} \text{ sau } \{1,1\}$$

în prezența unor defectări în sumator.

Pornind de la această soluție dată unui sistem autotestabil, se poate da o generalizare pentru cazul sistemelor care au k grupe de semnale de ieșire: fiecăreia dintre cele k grupe îi corespunde cîte un semnal de test, y_{T_i}. Pentru fiecare variabilă y_{T_i} se formează, cu circuitele modulului SR, variabila y'_{T_i} complementară lui y_{T_i} cu variabilele care reprezintă semnalele la intrările primare. Sistemul astfel sintetizat va avea $2k$ semnale de test. Ieșirile $\mathbf{Y}_F$ și $\mathbf{Y}_T$ pot fi codate, utilizîndu-se o codare separabilă cu un cod de paritate, cod Berger etc. Făcînd apel la schema din figura 3.15, se poate stabili că ieșirile $\mathbf{Y}_F$ ale modulului funcțional formează informația „utilă" a codului, în timp ce ieșirile $\mathbf{Y}_T$ ale modulului redondant formează informația „de control" a codului.

Această soluție propusă pentru sistemele autotestabile este rezervată sistemelor combinaționale, pentru care semnalele de ieșire depind numai de semnalele de intrare.

3.3.2.2. SISTEME AUTOTESTABILE CU STRUCTURĂ REDONDANTĂ NESEPARABILĂ

În această categorie pot fi incluse sistemele autotestabile sintetizate pe baza utilizării codurilor redondante de tip neseparabil pentru codarea informațiilor de intrare/ieșire, a funcțiilor de stare ale sistemului. Așa sînt, de exemplu, sistemele autotestabile realizate utilizînd codul k din n [11], [36].

Implementarea acestei structuri pentru sistemele autotestabile impune mai multe constrîngeri referitoare la concepția modulelor funcționale ale sistemului decît în cazul structurii cu redondanță separabilă. Pentru această

categorie de sisteme nu se poate identifica separat o mulțime a semnalelor funcționale și o mulțime a semnalelor de test. Evident, nu se va adăuga un decodor la ieșirea sistemului funcțional pentru a se codifica informația de ieșire, ci sistemul funcțional va avea inclusă, într-un mod neseparabil, redondanța necesară posibilității de testare a sistemului în funcționare normală.

Comparativ cu sistemele autotestabile cu structură dublă, sistemele realizate utilizînd aceste tehnici de implementare a autotestabilității vor avea un cost mai scăzut, deoarece includ mai puține circuite. Astfel, pentru a coda 2^m stări ale unui sistem secvențial sînt necesare — pentru sistemul neredondant, care nu este autotestabil — un număr de m variabile secundare; utilizarea structurii redondante dublate pentru implementarea autotestabilității echivalează cu necesitatea a $2m$ variabile secundare. Cu un cod k din n se pot coda un număr maxim de C_n^k stări. De exemplu, dacă $m = 4$, folosirea pentru codare a unui cod de tipul 3 din 6 — care evidențiază 20 configurații distincte — este suficient pentru a diferenția cele $2^4 = 16$ stări interne posibile. În cazul adoptării soluției de sistem autotestabil cu structură dublată sînt necesare 8 variabile secundare, iar cînd se folosește codul 3 din 6 pentru realizarea unui sistem autotestabil sînt necesare numai 6 variabile secundare, deci, mai puține circuite.

3.3.3. STRUCTURA CIRCUITELOR DE CONTROL

În § 3.3.1 au fost evidențiate rolul și locul unui circuit de control într-un sistem autotestabil, indicîndu-se totodată o definiție a circuitelor de control exprimată prin relațiile (3.17) și (3.18). Problema realizării circuitelor de control a fost abordată într-o serie de lucrări [2], [11], [23], [31] ș.a., fiind propuse metode de concepție a acestor circuite pentru controlul semnalelor de test care aparțin codurilor cu bit de paritate, codurilor k din n, codurilor Berger etc.

O soluție posibilă pentru realizarea unui circuit de control este aceea potrivit căreia semnalul de ieșire al circuitului este „0" atunci cînd sistemul controlat funcționează corect și „1" cînd apare o defectare în sistem. Dar, așa cum s-a reliefat în § 3.3.1, circuitul de control trebuie să fie el însuși autotestabil. Aceasta implică necesitatea ca numărul minim al ieșirilor unui circuit de control să fie cel puțin doi pentru a se evidenția și o informație de defectare în acesta.

Dacă circuitele de control se pot realiza cu circuite independente, adică fiecare semnal de ieșire este sintetizat de un singur circuit, care nu are nimic comun cu celelalte decît semnalele de intrare, o defectare într-un circuit va afecta doar o ieșire a circuitului de control. Există și o posibilitate simplă de aplicare a redondanței protective statice la nivelul acestor circuite, aceea ce va face ca semnalele de ieșire ale circuitului de control să nu fie afectate de respectiva defectare, iar circuitul de control să fie de înaltă fiabilitate.

În cele ce urmează se vor analiza unele probleme legate de realizarea circuitelor de control.

3.3.3.1. CONSTRUCȚIA CIRCUITELOR DE CONTROL PENTRU CODUL CU BIT DE PARITATE

Aplicarea acestei codări în realizarea sistemelor autotestabile este simplă și necostisitoare: un grup de $n-1$ variabile ale oricărei configurații binare este codat prin adăugarea unei variabile cu ajutorul operatorului SAU EXCLUSIV.

O mulțime de configurații binare, x_i, aparțin unui cod cu bit de paritate, dacă este verificată [23] relația:

$$x_1 \oplus x_2 \oplus \ldots \oplus x_n = 0 \qquad (3.19)$$

unde $\oplus$ reprezintă operatorul SAU EXCLUSIV. Prin urmare, circuitul de control, care intră în structura sistemelor autotestabile cu semnalele de ieșire codate utilizînd un astfel de cod, trebuie să efectueze o verificare a ecuației (3.19).

În realizarea acestor circuite de control se disting două cazuri corespunzătoare existenței sau inexistenței la intrările circuitului a tuturor combinațiilor care aparțin codului considerat.

1°. *La intrarea circuitului de control apar toate combinațiile.* În acest caz toate cuvintele posibile ale codului, scrise utilizînd cele $n-1$ variabile, sînt în număr de 2^{n-1}. Deci, la intrarea circuitului de control apar 2^{n-1} combinații de semnale binare.

În aceste condiții, o metodă de realizare a unui circuit de control [2] presupune următoarele etape:

- mulțimea variabilelor binare de intrare este arbitrar divizată în două grupe;
- fiecare grupă este asociată unei funcții care efectuează suma modulo 2 a tuturor variabilelor care aparțin grupei;
- pentru fiecare dintre aceste funcții se sintetizează cîte un circuit independent.

În figura 3.19 se dau două exemple de utilizare a metodei pentru realizarea circuitelor de control, cînd numărul semnalelor de intrare este $n = 8$.

Proprietatea de *autostabilitate* a circuitelor de control sintetizate după această metodă este garantată prin aplicarea semnalelor de intrare care baleiază în mod sigur cele 2^{n-1} combinații de semnale la intrările primare, necesare testării tuturor situațiilor posibile pentru controlul circuitelor SAU EXCLUSIV. Se știe că un circuit SAU EXCLUSIV cu k intrări necesită un număr de 2^k vectori-semnale de intrare pentru baleierea „tabelei de adevăr" a circuitului. Pe această bază în lucrarea [23] s-a demonstrat că numărul minimal de combinații ale semnalelor de intrare necesare unui circuit de control al codului cu bit de paritate, care utilizează module SAU EXCLUSIV cu k intrări, este 2^k. Dar orice submulțime de k variabile ale unui cod de paritate ia 2^k configurații distincte. Deci, orice realizare în arbore cu porți SAU EXCLUSIV după modelul celei prezentate în figura 3.19 va fi testată în timpul funcționării normale, cînd circuitul va primi cele 2^k configurații binare de intrare.

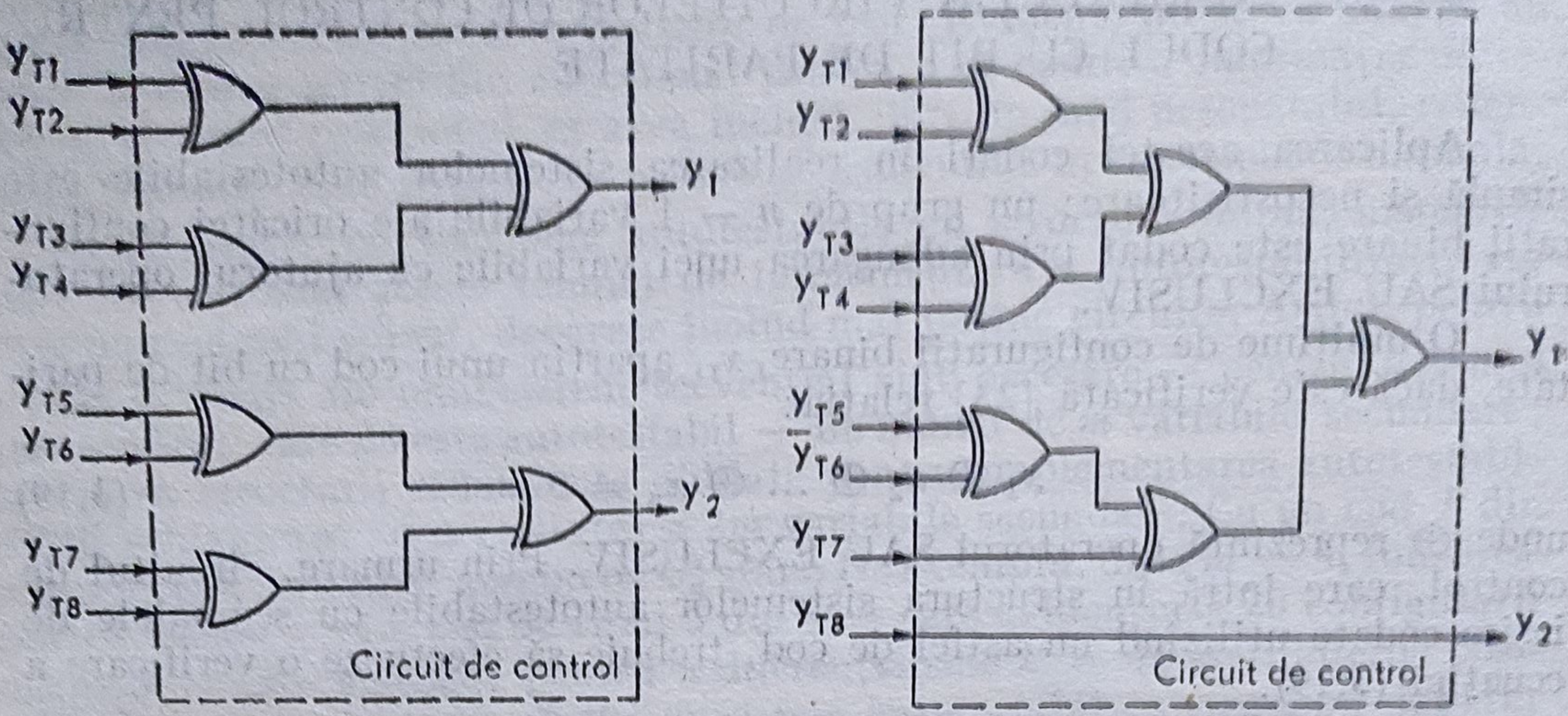


Fig. 3.19. Circuite de control pentru sistemele autotestabile implementate cu cod cu bit de paritate:

a — semnalele $y_{T_1}, \ldots, y_{T_8}$ au fost împărțite în două subunități, $y_{T_1}, \ldots, y_{T_4}$ și $y_{T_5}, \ldots, y_{T_8}$; *b* — semnalele $y_{T_1}, \ldots, y_{T_8}$ au fost împărțite în submulțimile $y_{T_1}, \ldots, y_{T_7}$ și y_{T_8}.

Un caz particular interesant este acela al unei rețele celulare formate din $n - 2$ porți SAU EXCLUSIV cu intrări conectate în cascadă (fig. 3.20). În continuare se va arăta că circuitul este el însuși autotestabil.

Se constată cu ușurință că dacă mulțimea celor patru configurații binare a semnalelor de intrare x_i, indicate mai jos, este prezentă la intrarea circuitului de control, atunci toate cele 2^2 configurații binare de intrare — 00, 01, 10, 11 — necesare testării unei porți SAU EXCLUSIV sînt prezente la intrarea acesteia. Astfel,

$$
\begin{aligned}
&(1)\ x_i = 0 \qquad \forall\, i = \{1, \ldots, n\};\\
&(2)\ x_i = 1 \qquad \forall\, i \in \{1, \ldots, n-1\};\\
&\qquad\qquad x_n = x_1 \oplus x_2 \oplus \ldots \oplus x_{n-1}\\
&(3)\ x_1 = 0,\ x_i = 1, \forall\, i\{2. \ldots, n-1\};\\
&\qquad\qquad x_n = x_1 \oplus x_2 \oplus \ldots \oplus x_{n-1}\\
&(4)\ x_1 = 1,\ x_i = 0, \forall i \in \{2, \ldots, n-1\};\\
&\qquad\qquad x_n = x_1 \oplus x_2 \oplus \ldots \oplus x_{n-1}
\end{aligned}
\qquad (3.20)
$$

2°. *La intrarea circuitului de control apare un număr incomplet de combinații de semnale.* Dacă la intrarea circuitului de control apare un număr incomplet de combinații ale semnalelor de intrare — deci, nu sînt baleiate toate cuvintele codului — obținerea unui circuit de control de tip autotestabil nu este întotdeauna posibilă.

Pe baza considerentelor de mai sus (relațiile 3.20), în cele ce urmează se prezintă un procedeu sistematic de cercetare a posibilității realizării unui circuit de control cu porți SAU EXCLUSIV, care primește la intrare semnale aparținînd unui cod cu bit de paritate, dar cu un număr incomplet de combinații [7].

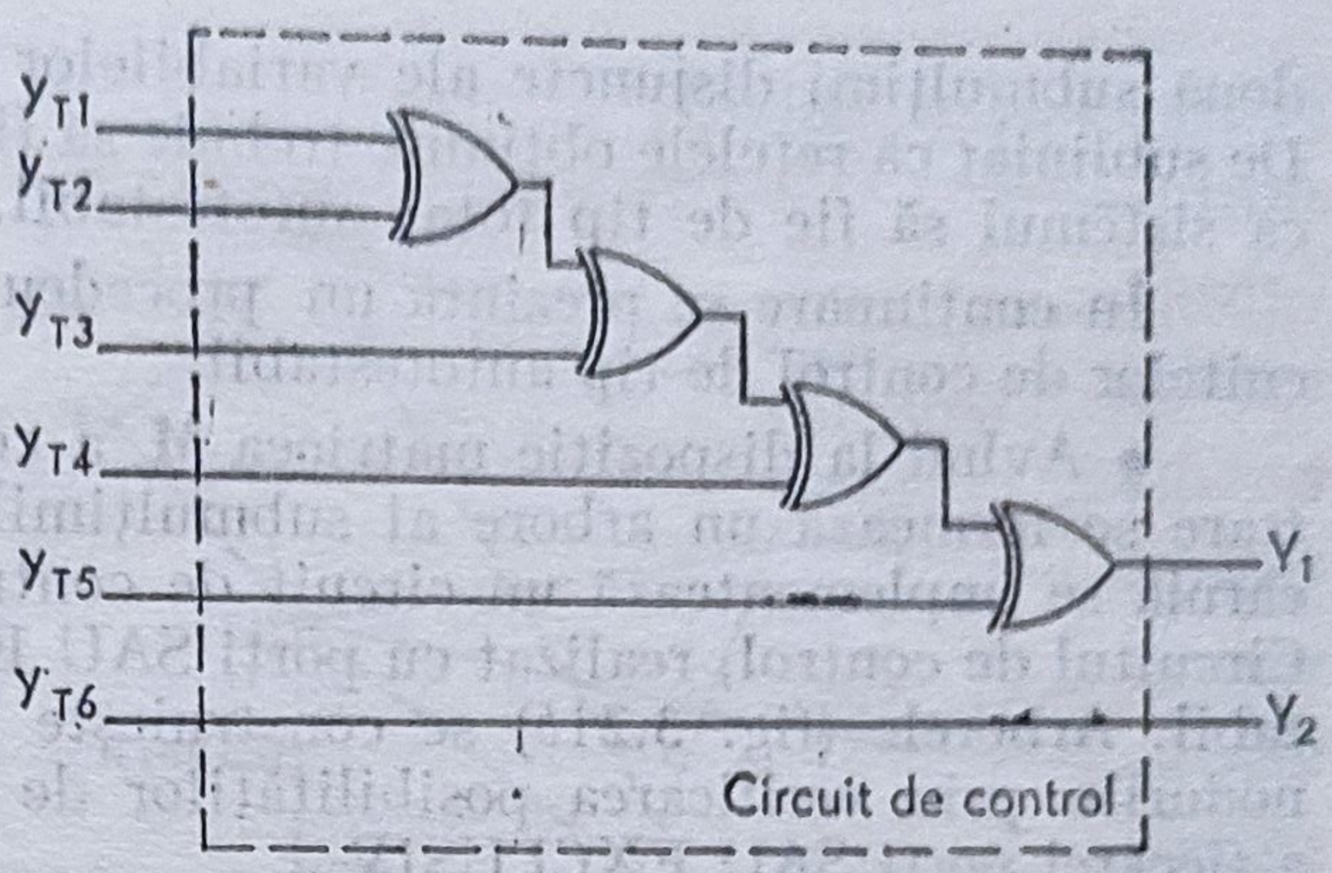


Fig. 3.20. Circuit de control autotestabil pentru codul cu bit de paritate cu porți SAU EXCLUSIV conectate în cascadă.

Se consideră în acest sens un circuit de control cu structura indicată în figura 3.20, implementat cu porți SAU EXCLUSIV.

• Fie $m < 2^{n-1}$ numărul de configurații binare de n variabile ale unui cod bit de paritate cu care se codifică semnalele de intrare în circuitul care urmează a fi sintetizat.

• Domeniul semnalelor de intrare ale acestui circuit de control se va prezenta într-o matrice, **M**, cu m linii și n coloane. Fiecare coloană evidențiază starea binară luată de variabila de intrare x_i corespunzătoare în cele m configurații binare prezente la intrare. Matricea **M** construită în acest mod va permite o verificare ușoară, pas cu pas, a rețelei celulare ce urmează a fi obținută.

Exemplul 3.4. Fie o mulțime de trei semnale de intrare, x_1, x_2, x_3, la intrarea unui circuit de control ce urmează a fi sintetizat. Mulțimea semnalelor de intrare ia doar cele cinci configurații, cuprinse în matricea **M**:

$$M = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}. \tag{3.21}$$

Dacă circuitul care realizează funcția $x_1 \oplus (x_2 \oplus x_3)$ este implementat cu porți SAU EXCLUSIV, este ușor de verificat cu aranjamentul dat în matricea **M** că fiecare poartă SAU EXCLUSIV primește la intrare semnalele 00, 01, 10 și 11; deci, rețeaua de porți obținută ca în figura 3.20, care realizează această funcție, este de tip autotestabil. În schimb, se constată, cu aranjamentul dat de matricea **M**, că realizarea $(x_1 \oplus x_2) \oplus x_3$ nu este de tip autotestabil, deoarece poarta SAU EXCLUSIV care realizează funcția $x_1 \oplus x_2$ nu va primi niciodată configurația 01 cu cele 5 configurații date semnalelor de intrare. De altfel, operatorul SAU EXCLUSIV nu îndeplinește proprietatea de asociativitate, adică

$$x_1 \oplus (x_2 \oplus x_3) \neq (x_1 \oplus x_2) \oplus x_3.$$

În exemplele anterioare a fost indicată soluția realizării a două rețele de porți SAU EXCLUSIV pentru sinteza funcțiilor Y_1 și Y_2 definite pe

două submulțimi disjuncte ale variabilelor x_i care verifică ecuația (3.19) De subliniat că rețelele obținute trebuie să fie ele însele autotestabile pentru ca sistemul să fie de tip total autotestabil.

În continuare se prezintă un procedeu sistematic de realizare a circuitelor de control de tip autotestabil.

• Avînd la dispoziție matricea **M** a combinațiilor semnalelor de intrare se formează un arbore al submulțimilor variabilelor x_i, cu ajutorul căruia se implementează un circuit de control care verifică ecuația (3.19). Circuitul de control, realizat cu porți SAU EXCLUSIV, este de tip autotestabil. Arborele (fig. 3.21*b*) se construiește de sus în jos, crescînd treptat nodurile prin verificarea posibilităților de testare în funcționare normală a fiecărei porți SAU EXCLUSIV.

• Algoritmul verificării autotestabilității reuniunii a două subrețele de porți SAU EXCLUSIV — implementarea a două submulțimi variabile x_i — constă în următoarele: pornind de la matricea **M** se cercetează treptat dacă configurațiile semnalelor binare 00, 01, 10 și 11 sînt prezente în funcționare normală la intrările fiecărei porți, pentru a apărea o indicație de defectare a rețelei de porți atunci cînd se defectează una dintre porți.

• Un nod al arborelui construit după acest procedeu reprezintă o rețea de porți SAU EXCLUSIV autotestabilă.

• Se va obține un circuit de control de tip autotestabil dacă se găsesc două substructuri ale arborelui definite pe două submulțimi complementare ale variabilelor de intrare.

E x e m p l u l 3. 5. Fie matricea **M** a variabilelor de intrare dată în figura 3.21*a*. Aceasta cuprinde — pe orizontală — cinci cuvinte scrise cu un cod cu bit de paritate de cinci variabile, reprezentînd configurațiile semnalelor binare prezente la intrarea circuitului de control care urmează a se realiza.

Pe baza metodei indicate mai sus se construiește arborele subrețelelor autotestabile. În practică, construirea unui asemenea arbore se oprește de îndată ce s-a obținut o soluție de două subrețele de porți logice autotestabile, avînd la intrare semnalele împărțite în două mulțimi complementare. Arborele rezultat este indicat în figura 3.21*b*. În dreptul nodurilor arborelui sînt trecute subrețelele obținute prin aplicarea operatorului SAU EXCLUSIV de exemplu, prin simbolul (1 2) 3 este indicată funcția $(x_1 \oplus x_2) \oplus x_3$ etc.

Din explorarea nodurilor arborelui se obține mulțimea soluțiilor circuitelor de control, prin identificarea perechilor de noduri care acoperă toate variabilele de intrare. În acest exemplu s-au găsit 19 soluții la nivelul celui de-al patrulea nod și 7 soluții prin compunerea subrețelelor la nivelul celui de-al treilea și al doilea nod. În figurile 3.22*a* și 3.22*b* sînt date două dintre cele 19 soluții, iar în figura 3.22*c* — una dintre cele 7 soluții.

În vederea obținerii unei soluții de tipul celei indicate în figura 3.22*b*, este suficientă elaborarea unui arbore redus începînd cu scrierea nodului corespunzător lui x_1, iar apoi a celorlalte noduri corespunzătoare lui $x_1 \oplus x_2$, $(x_1 \oplus x_2) \oplus x_3$ etc.; se verifică pe rînd, din aproape în aproape, autotestabilitatea subrețelelor de porți SAU EXCLUSIV rezultate iar nodurile care nu au succesori nu se mai dezvoltă. Procedîndu-se astfel într-o tratare

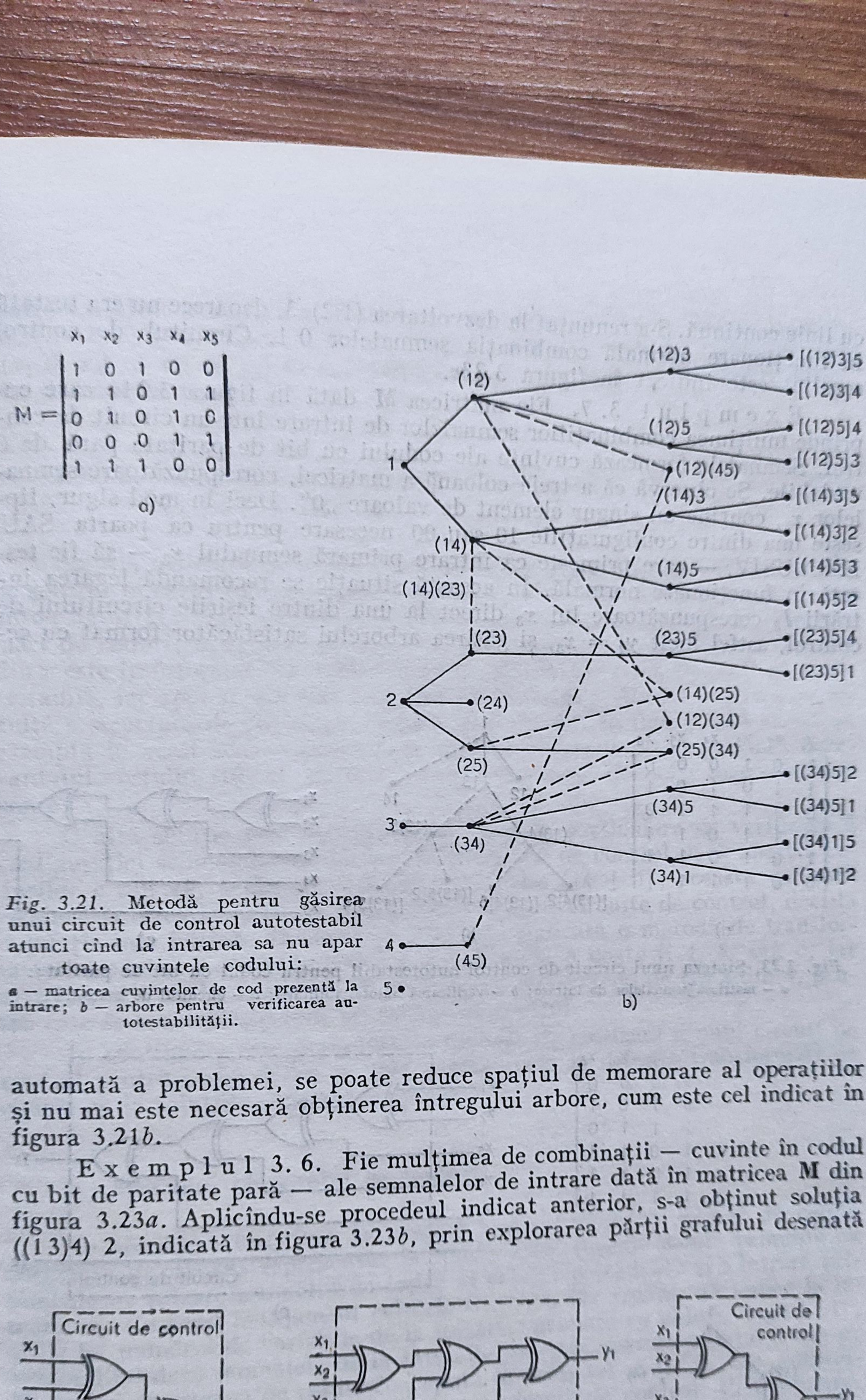


Fig. 3.21. Metodă pentru găsirea unui circuit de control autotestabil atunci cînd la intrarea sa nu apar toate cuvintele codului:

a — matricea cuvintelor de cod prezentă la intrare; *b* — arbore pentru verificarea autotestabilității.

automată a problemei, se poate reduce spațiul de memorare al operațiilor și nu mai este necesară obținerea întregului arbore, cum este cel indicat în figura 3.21*b*.

E x e m p l u l 3. 6. Fie mulțimea de combinații — cuvinte în codul cu bit de paritate pară — ale semnalelor de intrare dată în matricea **M** din figura 3.23*a*. Aplicîndu-se procedeul indicat anterior, s-a obținut soluția ((1 3)4) 2, indicată în figura 3.23*b*, prin explorarea părții grafului desenată

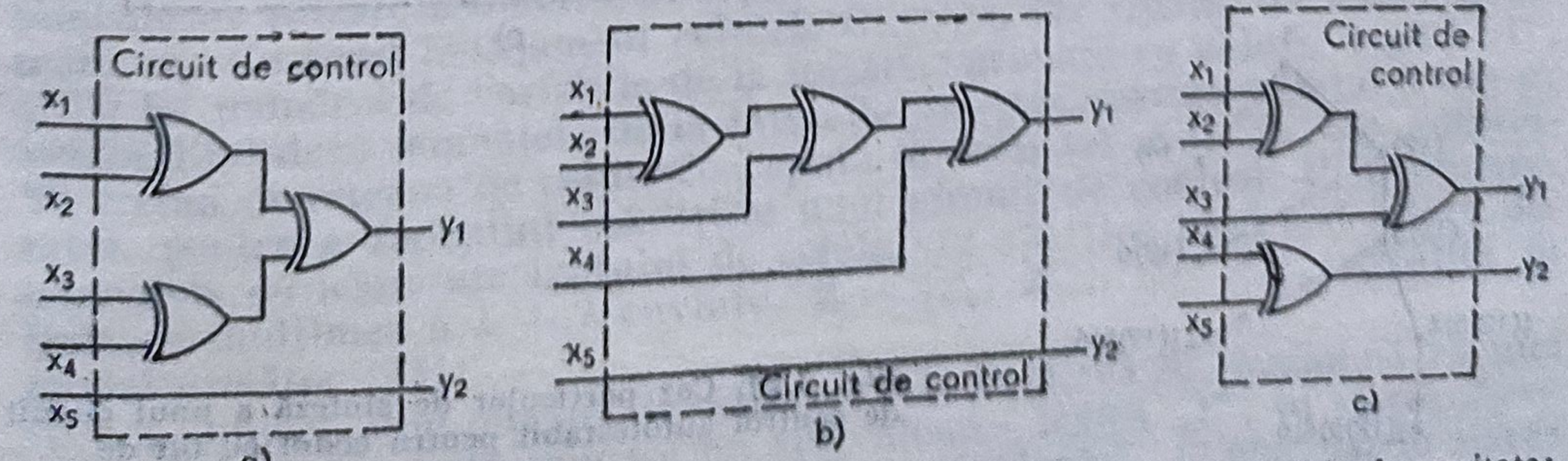


Fig. 3.22. Exemple de circuite de control autotestabile pentru codul cu bit de paritate: *a* — gruparea $(x_1 + x_2) + (x_3 + x_4)$ și x_5; *b* — gruparea $((x_1 + x_2) + x_3) + x_4)$ și x_5; *c* — gruparea $((x_1 + x_2) + x_3)$ și $(x_4 + x_5)$.

cu linie continuă. S-a renunțat la dezvoltarea (1 2) 3, deoarece nu era testată în funcționare normală combinația semnalelor 0 1. Circuitul de control rezultat este indicat în figura 3.23*c*.

E x e m p l u l 3. 7. Fie matricea **M** dată în figura 3.24*a* care cuprinde mulțimea combinațiilor semnalelor de intrare într-un circuit de control. Semnalele formează cuvinte ale codului cu bit de paritate pară de 6 variabile. Se observă că a treia coloană a matricei, corespunzătoare semnalelor x_3, conține un singur element de valoare „0". Deci în mod sigur, lipsește una dintre configurațiile 10 sau 00 necesare pentru ca poarta SAU EXCLUSIV, — care primește ca intrare primară semnalul x_3 — să fie testată în funcționare normală. În această situație se recomandă legarea intrării I_3 corespunzătoare lui x_3 direct la una dintre ieșirile circuitului de control, astfel încît $y_2 = x_3$, și găsirea arborelui satisfăcător format cu ce-

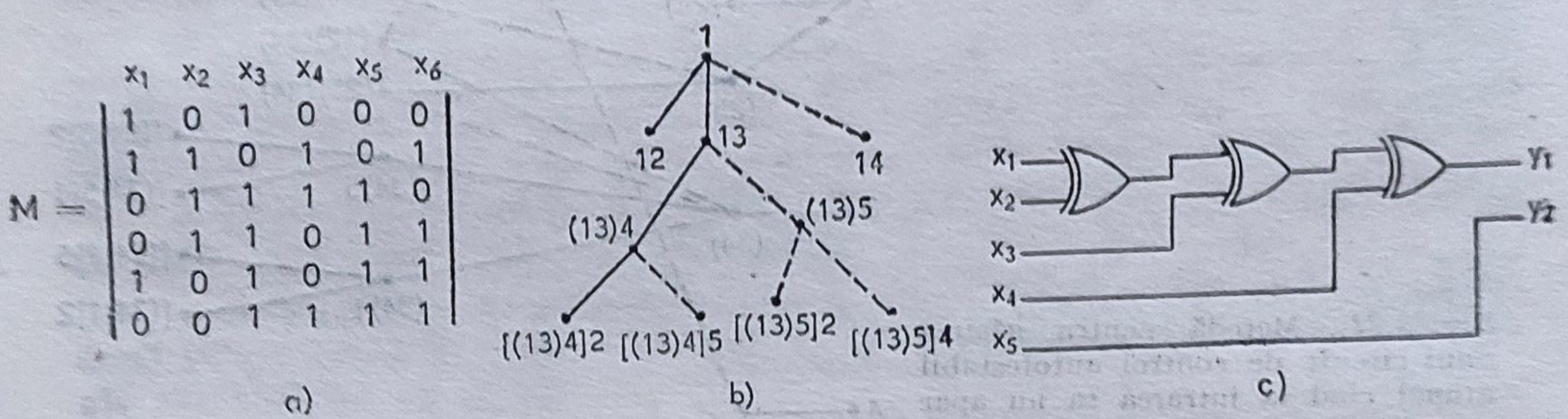


Fig. 3.23. Sinteza unui circuit de control autotestabil pentru codul cu bit de paritate:
a — matricea semnalelor de intrare; *b* — verificarea autotestabilității; *c* — circuitul de control.

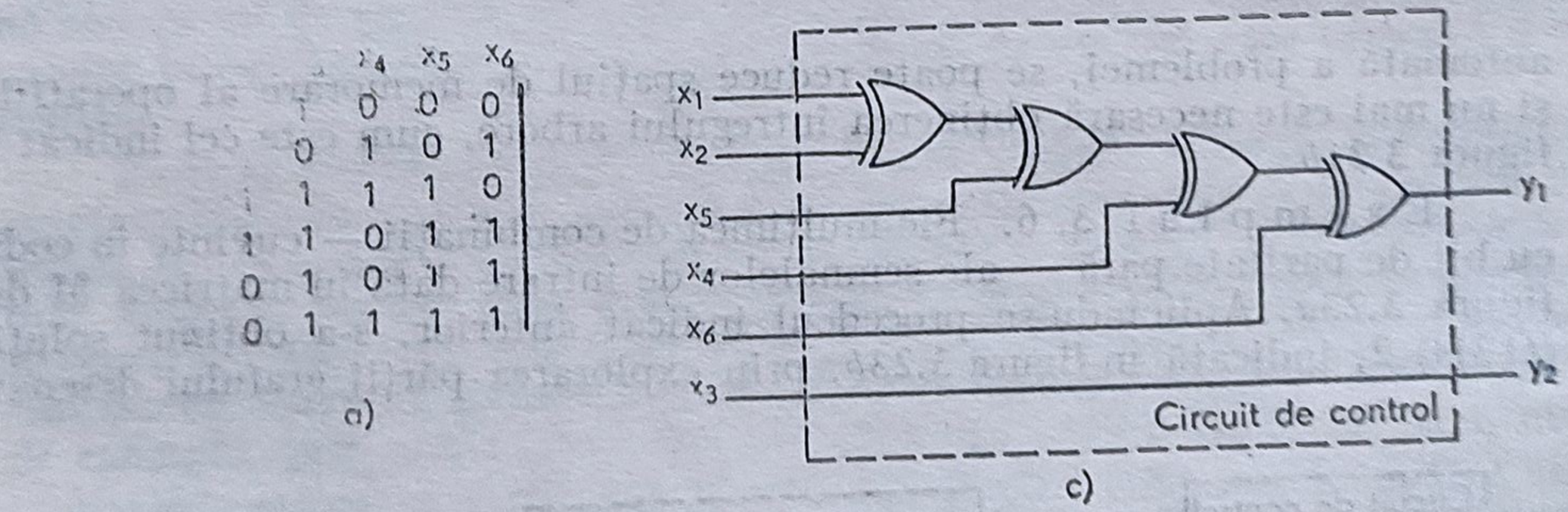


Fig. 3.24. Caz particular de sinteză a unui circuit de control autotestabil pentru codul cu bit de paritate:
a — matricea semnalelor de intrare; *b* — verificarea autotestabilității; *c* circuitul de control.

delalte variabile x pentru a se sintetiza semnalul y_1. În acest caz arborele este (((1 2) 5) 4) 6, indicat în figura 3.24*b*, iar circuitul care realizează funcția $(((x_1 \oplus x_2) \oplus x_4) \oplus x_5) \oplus x_6$, implementat cu porți SAU EXCLUSIV, este desenat în figura 3.24*c*.

3.3.3.2. CONSTRUCȚIA UNUI CIRCUIT DE CONTROL PENTRU CODUL k DIN n

Soluția utilizată de majoritatea autorilor pentru realizarea circuitelor de control în cazul folosirii codului k din n în codarea funcțiilor sistemului autotestabil [1] [11], [31] etc. constă în efectuarea unor serii de transformări de coduri, utilizîndu-se circuite autotestabile. Astfel, inițial codul k din n este transformat în codul 1 din C_n^k utilizînd o rețea de porți ȘI autotestabilă, iar apoi acesta este transformat în codul p din $2p$ pentru care rezultă o structură de circuit de control mai simplă. În figura 3.25 se dă un exemplu în acest sens. Circuitul de control conceput astfel prezintă dezavantajul costului ridicat, datorită numărului mare de porți logice componente.

O altă soluție este utilizabilă în cazurile particulare de verificare a configurației semnalelor de intrare în circuitele de control dacă aparțin codurilor k din $2k$, k din $2k + 1$ și $(k + 1)$ din $(2k + 1)$; această variantă conduce la o realizare pe două niveluri a acestor circuite de control, deci la o soluție mai ieftină. În lucrarea [31] este indicată o metodă de transformare a codului k din n în codul 1 din p, unde p poate fi 4, 5, sau 6, iar ulterior, utilizîndu-se o rețea de porți SAU, într-un cod de tip 2 din 4, pentru care se construiește ușor un circuit de control.

În continuare este dezvoltată o metodă de realizare a unui circuit de control pentru codul k din n, fără a mai apela la diferite transformări de coduri ca în soluțiile indicate anterior.

A. Structura circuitului de control

Se consideră circuitul de control realizat dintr-un număr de celule conectate în cascadă reprezentat în figura 3.26. Fiecare celulă primește ca semnale de intrare semnalul de ieșire al celulei precedente și o intrare primară, x_i. Cu acest aranjament celulele $C_1,\ldots,C_n$ vor trebui să indice la ieșirile lor numărul de variabile de la intrările primare cu valoare logică „1", evidențiind dacă semnalele de la intrările primare aparțin codului k din n. Va trebui ca rețeaua de celule concepută în acest fel să fie de tip autotestabil, pentru a îndeplini condițiile unui circuit de control. Prin urmare, semnalele de ieșire ale lanțului de celule vor aparține unui cod intern definit pe mulțimea a $k + 2$ cuvinte, $\mathbf{Z} = \{Z_0, Z_1,\ldots,Z_k, Z_e\}$, stabilite în modul următor [23]:

• Atunci cînd rețeaua de celule C_i este în starea Z_0, înseamnă că nici un semnal la intrările primare nu are valoarea logică „1"; în starea Z_1 — un semnal la o intrare primară are valoarea logică „1" etc., iar în starea Z_e — la mai multe de k intrări primare semnalele de intrare sînt de valoare

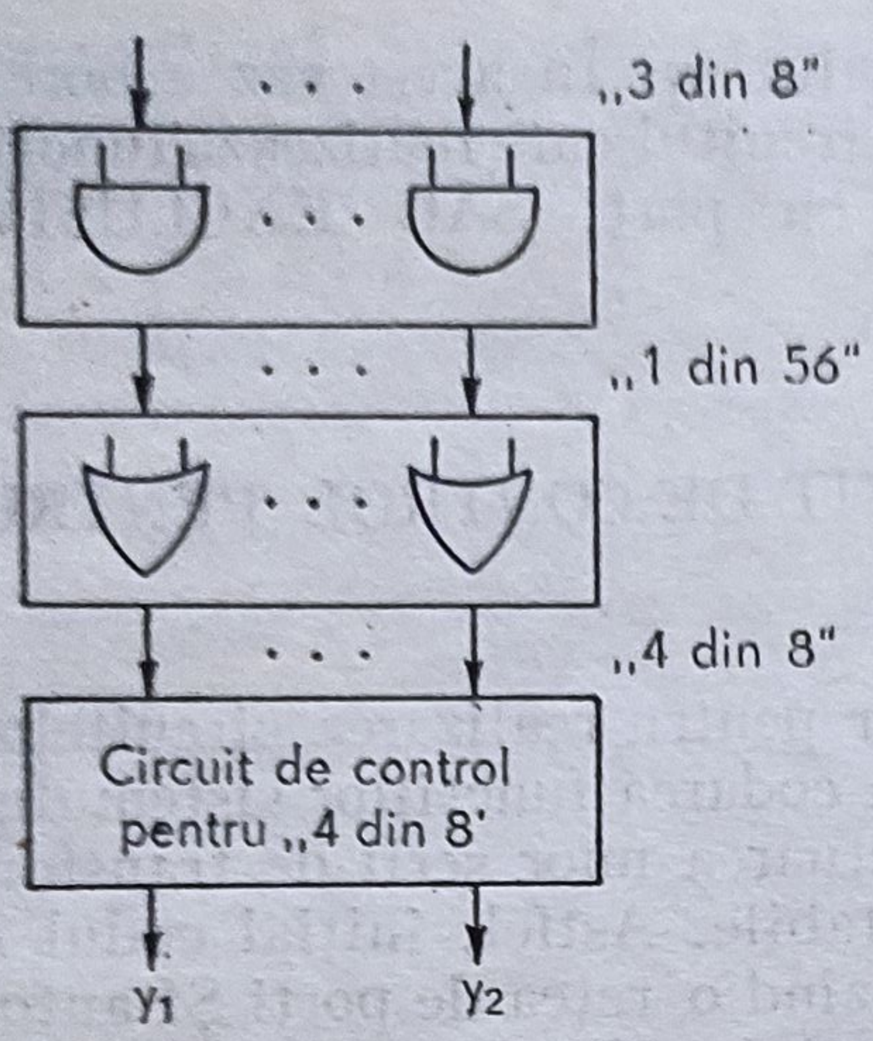


Fig. 3.25. Circuit de control autotestabil pentru codul k din n (exemplificare pentru codul 3 din 8).

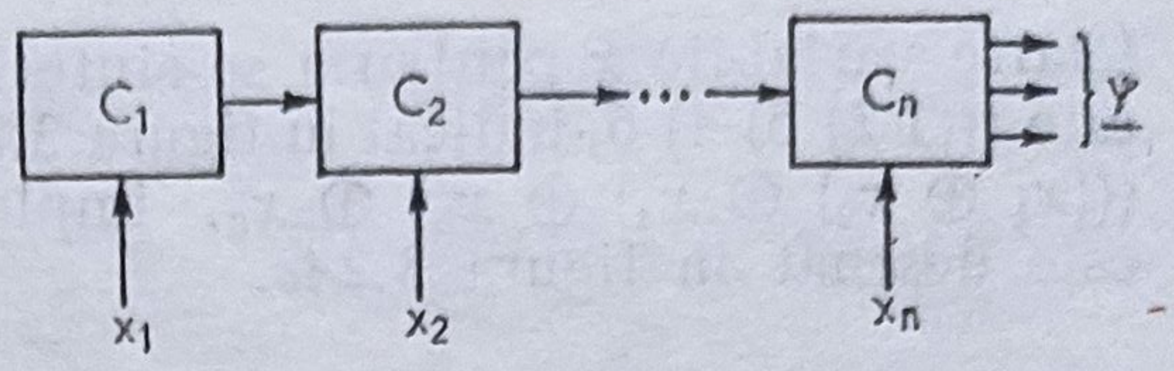


Fig. 3.26. Circuit de control autotestabil cu structură celulară pentru codul k din n.

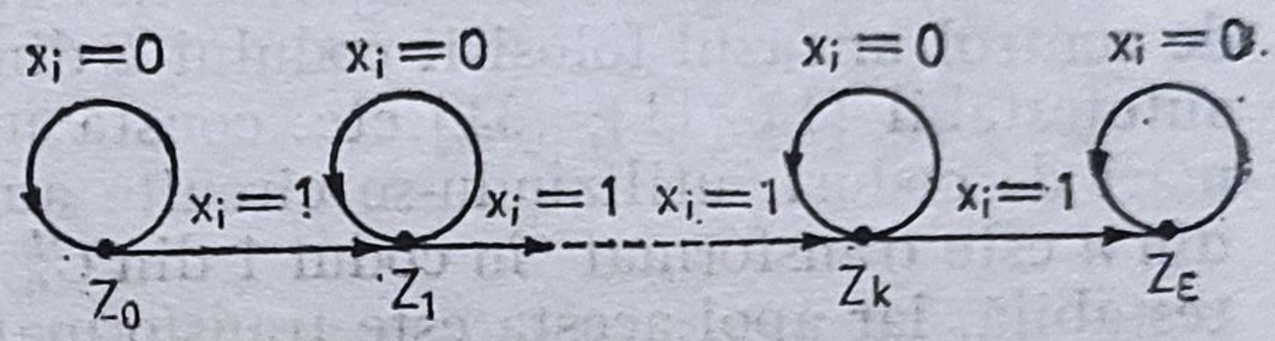


Fig. 3.27. Graful tranzițiilor stărilor circuitului de control pentru codul k din n.

logică „1". Graful tranzițiilor stărilor sistemului astfel conceput este indicat în figura 3.27. Pe graf sînt evidențiate tranzițiile posibile și anume:

$$\begin{aligned} &- \text{dacă } x_i = 0,\ Z_j \to Z_j,\ \forall\, j \in (0,1,\ldots k,) \\ &- \text{dacă } x_i = 1,\ \begin{cases} Z_j \to Z_{j+1},\ \forall\, \mathrm{j} < k \\ Z_j \to Z_\varepsilon,\ \forall\, j \in \{k,\ \varepsilon\} \end{cases} \end{aligned} \tag{3.22}$$

• De menționat că pentru primele k celule tranzițiile sînt mai simple. Astfel, celula C_1 realizează doar configurația Z_0 sau Z_1, după cum semnalul de intrare, x_1, ia valoarea logică „0" sau „1". Celula C_2 realizează doar configurațiile Z_0, Z_1 sau Z_2. Astfel, configurația Z_0 este obținută atunci cînd semnalul x_2 este „0", iar configurația realizată de C_1 este Z_0; configurația Z_1 este obținută cînd semnalul x_2 este „0", iar configurația realizată de C_1 este Z_1, sau cînd semnalul x_2 este „1", iar configurația realizată de C_1 este Z_0; configurația Z_2 este obținută atunci cînd semnalul x_2 este „1", iar configurația realizată de celula C_1 este Z_1. Celula C_3 realizează doar configurațiile Z_0, Z_1, Z_2 sau Z_3 ș.a.m.d. Acest comportament corespunde unui numărător, numai că informațiile „1" numărate sînt dispuse topologic și nu secvențial ca la numărătoarele secvențiale.

După cum s-a evidențiat, atunci cînd configurația semnalelor de intrare, x_i, aparține codului k din n, ultima celulă C_n va trebui să realizeze configurația Z_k. Dimpotrivă, dacă numărul de biți „1" din configurația semnalelor de intrare este diferit de k, ieșirea celului C_n va evidenția configurația Z_i, $i < k$, sau Z_ε, cu condiția că numărul biților „1" să fie mai mare decît k.

Din analiza funcționării acestui model rezultă că numărul maxim de ieșiri ale unei celule C_i trebuie să fie $k + 1$ pentru a evidenția cele $k + 2$ configurații Z_j. De aceea devine necesar ca stările Z_0, Z_1,...,Z_k, Z_ε să fie codificate după cum urmează:

$$|Z_0, Z_1, Z_2, \ldots, Z_{k-1}, Z_k, Z_\varepsilon| = \left|\begin{array}{ccccccc|c} \text{stare} & & & & & & & \text{ieşiri} \\ Z_0 & Z_1 & Z_2 & \ldots & Z_{k-1} & Z_k & Z_\varepsilon & \\ 0 & 1 & 1 & \ldots & 1 & 1 & 1 & y_i^0 \\ 1 & 0 & 1 & \ldots & 1 & 1 & 1 & y_i^1 \\ 1 & 1 & 0 & \ldots & 1 & 1 & 1 & y_i^2 \\ \cdot & \cdot & \cdot & \ldots & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \ldots & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \ldots & \cdot & \cdot & \cdot & \cdot \\ 1 & 1 & 1 & \ldots & 0 & 1 & 1 & y_i^{k-1} \\ 1 & 1 & 1 & \ldots & 1 & 0 & 1 & y_i^k \end{array}\right| \qquad (3.23)$$

Cele $k + 1$ semnale de la ieşirile unei celule C_i au fost notate în relaţia de definiţie (3.23) cu simbolurile y_i^0, $y_i^1 \ldots y_i^{k-1}$, y_i^k.

Pentru ca sistemul astfel conceput să fie un *circuit de control* el trebuie să fie de tip autotestabil. Din acest motiv este necesar ca celulele C_i să fie realizate cu circuite combinaţionale independente, iar fiecare funcţie de ieşire, y_i^j, să fie realizată de un astfel de circuit. De asemenea este necesar ca ieşirile acestor circuite să ia în cursul funcţionării normale cel puţin o dată valoarea logică „1" şi cel puţin o dată valoarea logică „0" pentru a fi îndeplinită condiţia de autotestabilitate.

Pe baza analizei tranziţiilor celulelor C_i şi avîndu-se în vedere modul de definire a configuraţiilor Z_j, dat de matricea (3.23), rezultă că pentru celula C_1 semnalele de ieşire y_1^2, y_1^3, ..., y_1^k vor fi întotdeauna „1", după cum şi semnalele y_2^3, y_2^4, ..., y_2^k vor fi întotdeauna „1" etc., sau semnalele y_{n-k+1}^0, y_{n-k+2}^0, y_{n-k+1}^1 vor fi de asemenea „1". În consecinţă, condiţia de autotestabilitate impusă permite eliminarea tuturor acestor variabile care, în cursul funcţionării normale nu iau decît o singură valoare, ceea ce înseamnă şi o minimizare a structurii acestor celule.

Schema-bloc a circuitului de control conceput în acest fel este detaliată în figura 3.28. Se constată astfel că celula C_n din figura 3.26 a dispărut. Semnalele de ieşire ale circuitului de control vor fi y_{n-1}^{k-1}, y_{n-1}^k şi y_n^k, ultimul coincizînd cu semnalul x_n,

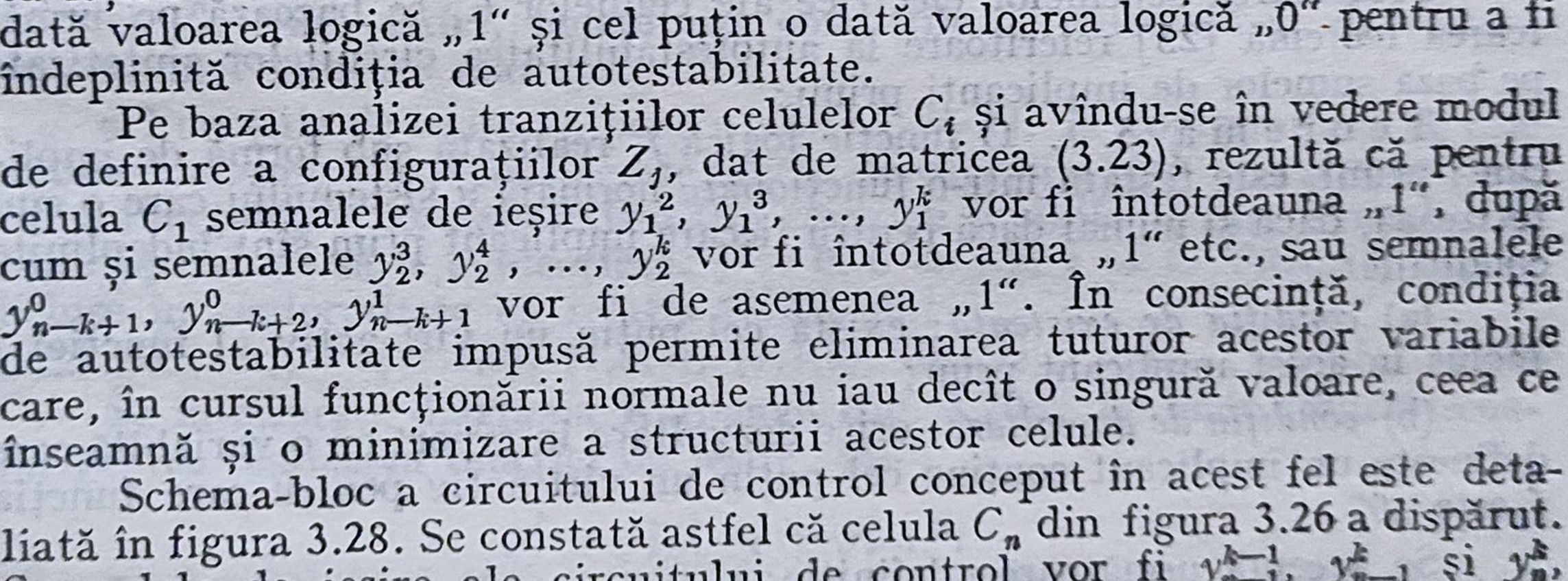


Fig. 3.28. Schema-bloc a circuitului de control autotestabil cu structură celulară pentru codul k din n.

Conform grafului tranziţiilor indicat în figura 3.27, aceste trei ieşiri ale circuitului de control vor lua în funcţionare normală două combinaţii de valori, astfel încît codul de ieşire al circuitului de control proiectat în acest fel va fi:

$$CE_c = \{y_{n-1}^{k-1},\ y_{n-1}^{k},\ x_n\} = \{(100),\ (0,11\}. \tag{3.24}$$

Ieşirile circuitului au fost stabilite astfel încît să fie îndeplinite relaţiile de definiţie ale unui circuit de control (3.17) şi (3.18).

B. Structura celulelor C_i

Structura fiecărei celule C_i trebuie stabilită astfel încît reţeaua de celule să sintetizeze semnalele indicate mai sus şi să fie ea însăşi autotestabilă. Prin urmare, ar trebui ca oricare defectare care va afecta o celulă C_i să fie detectată la ieşirea celulei respective şi, totodată, defectările individuale ale fiecărei celule să se propage la ieşirea sistemului de celule.

În soluţia adoptată aici, se disting trei tipuri de celule C_i, după cum $i \leqslant k$, $k+1 \leqslant i \leqslant n-k$ sau $i > n-k$. Anterior s-a propus ca fiecare dintre aceste celule să fie realizată cu circuite logice combinaţionale. independente în scopul unui control mai facil al autotestabilităţii lor.

Proiectarea schemelor electrice ale celulelor C_i se bazează pe următoarea teoremă [23] referitoare la autotestabilitatea circuitelor sintetizate pe baza sumelor de implicanţi primi:

T e o r e m a 3. 1. O funcţie logică f, realizată sub formă de implicanţi primi, este testată într-o funcţionare normală a sistemului care o sintetizează, dacă şi numai dacă pentru fiecare implicant prim sînt îndeplinite condiţiile:

(a) *condiţia de existenţă*: există cel puţin un vîrf activ al funcţiei f inclus în acest implicant prim;

(b) *condiţia de dezvoltare*: orice dezvoltare în raport cu o variabilă care defineşte un implicant conţine cel puţin un vîrf activ pentru care funcţia f este „0".

Se acceptă următoarea definiţie:

Fie o funcţie logică f de n variabile x_i, realizată de un circuit combinaţional; se numeşte *vîrf activ* orice configuraţie a variabilelor x_i care constituie semnalele de intrare în circuit, efectiv aplicate în funcţionare normală (configuraţia aparţine codului de intrare).

1°. *Cazul celulelor* C_i, $i \in [k+1,\ n-k]$. Fiecare celulă C_i are $k+1$ intrări, provenind de la celulele precedente (fig. 3.28), şi o intrare primară. Dacă la intrarea circuitului de control apar toate configuraţiile codului k din n controlat, atunci celula C_i primeşte în funcţionare normală ca semnale de intrare oricare dintre semnalele $(Z_j,\ x_i)$ în configuraţia:

$$\text{şi} \quad \left.\begin{array}{l} \{Z_j,\ 0\},\ \forall_j \in [0,\ k] \\ \{Z_j,\ 1\},\ \forall_j \in [0, k-1]. \end{array}\right\} \tag{3.25}$$

Mulţimea ieşirilor de la celula precedentă care vor fi intrări în această celulă constituie componentele vectorului Z_j. Combinaţia de semnale $\{Z_j,\ 0\}$

apare la intrarea celulei C_i dacă j semnale x_i la intrările primare ale celor j celule în amonte de celula C_i iau valoarea „1", iar intrarea primară a celulei C_i este „0" ș.a.m.d. Aranjamentul dat de expresia (3.25) reprezintă mulțimea vîrfurilor active ale funcției logice realizată de celula C_i.

• Semnalul de ieșire y_i^0 al celulei C_i este o funcție de semnalul y_{i-1}^0 — — de la ieșirea celulei C_{i-1} — și semnalul x_i, prezent la intrarea primară. Atunci cînd se realizează starea Z_0, semnalul de ieșire y_i^0 este „0" (matricea 3.24), iar celelalte semnale y_i^j vor fi „1". Pentru celelalte stări Z_j ale celulei C_i semnalul y_i^0 va fi „1" și un alt semnal y_i^j va fi „0". Pentru funcția de ieșire y_i^0 realizată de celula C_i, în conformitate cu relațiile (3.25), există următoarele vîrfuri active:

$$\{y_{i-1}^0,\ x_i\} = \{(0,\ 0)\ (0,\ 1),\ (1,0\ (1,1)\}. \tag{3.26}$$

Pe baza matricei de definiție a configurațiilor Z_j, dată de (3.23), funcția y_i^0 va fi „1" cînd variabilele y_{i-1}^0 și x_i sînt „0". Se întocmește diagrama Karnaugh corespunzătoare (fig. 3.29*a*) și prin minimizare se obține funcția y_i^0:

$$y_i^0 = x_i + y_{i-1}^0. \tag{3.27}$$

În această diagramă vîrfurile active sînt încadrate în pătrate.

Se observă că realizarea neredondantă dată de ecuația (3.27) este testată în funcționare normală: motivul constă în aceea că fiecare implicant prim conține un vîrf activ, iar $\{0,\ 0\}$ este de asemenea un vîrf activ al funcției y_i^0. Sînt astfel, îndeplinite condițiile de existență și dezvoltare ale teoremei indicate mai sus.

• Funcțiile de ieșire y_i^j, pentru $j \in [1,\ k-1]$, în conformitate cu cele arătate mai sus, vor fi considerate funcții de variabile y_{i-1}^{j-1}, y_{i-1}^j și x_i întrucît într-o funcționare normală există trei tranziții posibile pe care semnalul de ieșire y_i^j trebuie să le evidențieze: $1 \to 0$, $0 \to 1$ și $1 \to 1$. Vîrfurile active deduse în conformitate cu relația (3.25) sînt:

$$\{y_{i-1}^{j-1},\ y_{i-1}^j,\ x_i\} = \{(1,1,0),\ (1,1,1),\ (0,1,0),\ (0,1,1),(1,0,0,),\ (1,0,1)\}. \tag{3.28}$$

Se întocmește diagrama Karnaugh (fig. 3.29*b*), de unde rezultă ecuația pentru realizarea funcției de ieșire, y_i^j.

$$y_i^j = y_{i-1}^{j-1}\ x_i + y_{i-1}^j\ \bar{x}_i. \tag{3.29}$$

Se observă că realizarea dată de ecuația (3.29) pentru funcția de ieșire y_i^j este de tip autotestabil, deoarece sînt îndeplinite condițiile de *existență* și *dezvoltare* ale teoremei 3.1.

• Funcția de ieșire y_i^k va fi de asemenea o funcție de variabilele y_{i-1}^{k-1}, y_{i-1}^k și x_i. Implementarea acestei funcții este realizată de asemenea de ecuația (3.29), căci y_i^k va fi „1" în aceleași situații ca și y_i^j. Pentru această funcție, în conformitate cu relația de definiție (3.23), vîrfurile active sînt:

$$\{y_{i-1}^{k-1},\ y_{i-1}^k,\ x_i\} = \{(1,1,0),\ (0,1,0),\ (0,1,1,),\ (1,0,0),\ (1,0,1)\}; \tag{3.30}$$

analizîndu-se condițiile teoremei 3.1 se observă că realizarea dată funcției y_i^k de ecuația (3.29) va fi de asemenea de tip autotestabil și pentru acest caz.

În figura 3.30 se dă un exemplu de celulă astfel sintetizată pentru cazul $k = 3$, $n = 8$.

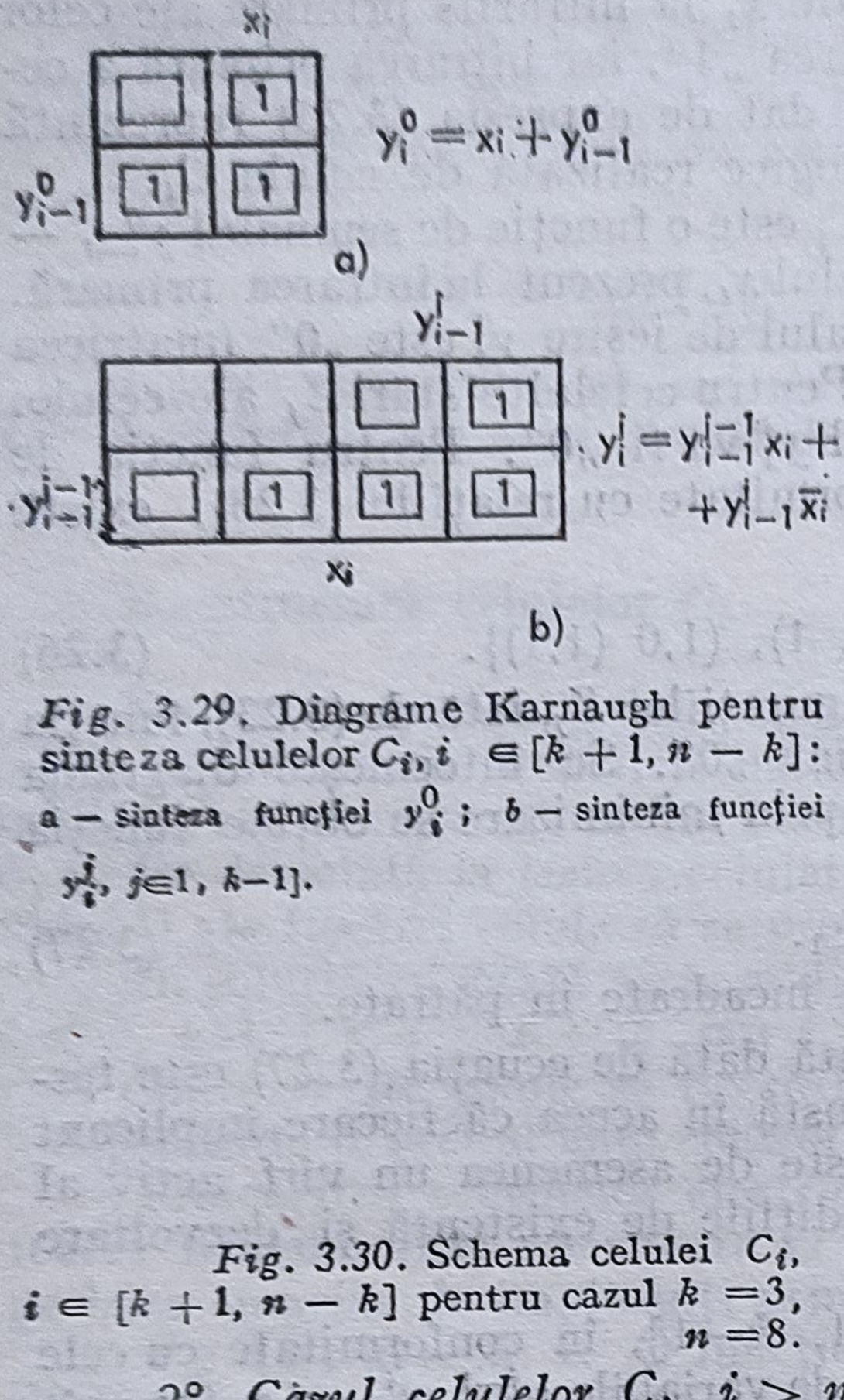


Fig. 3.29. Diagrame Karnaugh pentru sinteza celulelor C_i, $i \in [k+1, n-k]$:
a — sinteza funcției y_i^0; *b* — sinteza funcției y_i^j, $j \in [1, k-1]$.

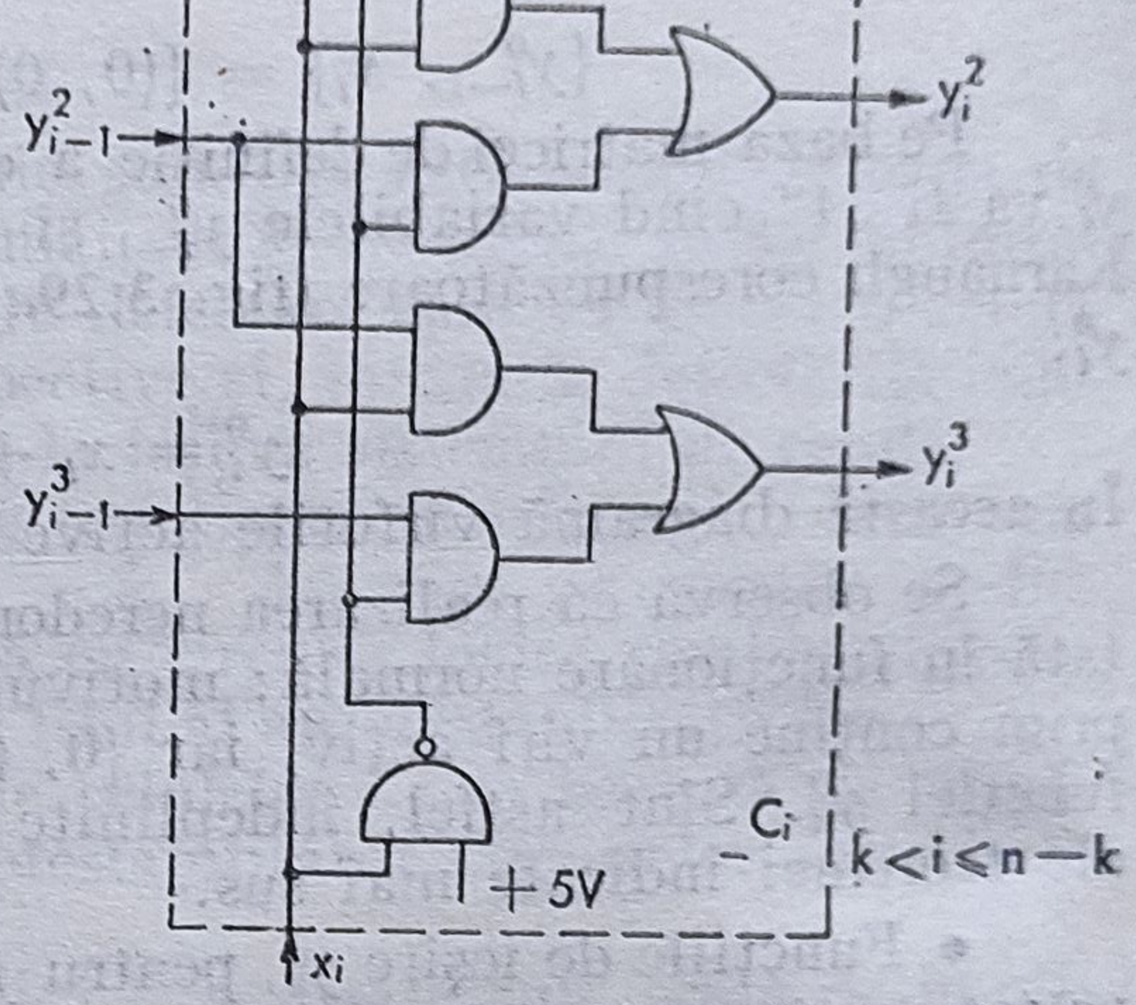


Fig. 3.30. Schema celulei C_i, $i \in [k+1, n-k]$ pentru cazul $k=3$, $n=8$.

2°. *Cazul celulelor* C_i, $i > n-k$. Comparativ cu celulele analizate aceste celule au un număr mai mic de ieșiri (fig. 3.28). Astfel, pentru celula C_{n-k+1} semnalul de ieșire y_{n-k+1}^0 nu este necesar, deoarece va fi întotdeauna „1"; de asemenea, nici pentru celula C_{n-k+2} nu sînt necesare semnalele de ieșire y_{n-k+2}^0, y_{n-k+2}^0 deoarece ele nu sînt purtătoare de informație ș.a.m.d.

Analizîndu-se numărul de intrări în fiecare celulă, se constată că:

— pentru celula C_{n-k+1} vor fi $(k+1)$ intrări de la celula precedentă și o intrare primară;

— pentru celula C_{n-k+2} vor fi k intrări — ieșirile celulei C_{n-k+1} — și o intrare primară ș.a.m.d.

De menționat că pentru aceste celule configurația $(Z_0, 0)$, nu va apărea niciodată într-o funcționare normală; în caz contrar ar însemna că există mai puțin de k biți „1" în vectorul semnalelor de intrare al circuitului de control căutat. De altfel, acesta a fost raționamentul pe baza căruia s-au suprimat din structura celulei C_{n-k+1} semnalul de ieșire y_{n-k+1}.

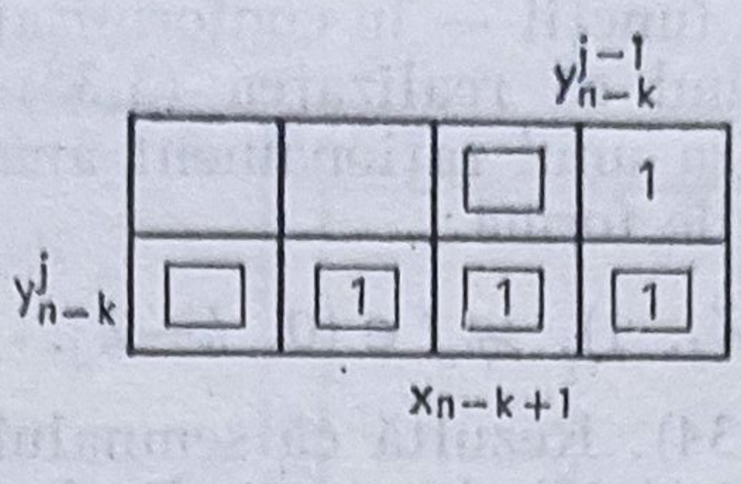
Celula C_{n-k+1} va primi la intrări, în funcționare normală, una dintre configurațiile:

$$(Z_j, 0), \forall j \in [1, k]; \qquad (3.31)$$
$$(Z_j, 1) \forall j \in [0, k-1].$$

Funcțiile de ieșire y_{n-k+1}^j, $j \geqslant 1$, vor fi definite pe mulțimea $\{y_{n-k}^{j-1}, y_{n-k}^j, x_{n-k+1}\}$, din aceleași considerente ca în cazul celulelor C_i, $i \in [k+1, n-k]$, cu valorile indicate în diagrama Karnaugh corespunzătoare din figura 3.31a. De aici se obține următoarea ecuație de implementare:

$$y_{n-k+1}^j = y_{n-k}^j \bar{x}_{n-k+1} + y_{n-k}^{j-1} x_{n-k+1}. \qquad (3.32)$$

Fig. 3.31. Diagrama Karnaugh pentru sinteza celulelor C_i, $i \in (n-k, n)$.



Pentru acest caz vîrfurile active sînt:

$$\{y^{j-1}_{n-k},\ y^{j}_{n-k},\ x_{n-k+1}\} = \{(1,0,0),\ (1,1,0),\ (0,1,1,),\ (1,0,1),\ (1,1,1)\},\quad (3.33)$$

mulțime care asigură îndeplinirea condițiilor de existență și dezvoltare ale teoremei 3.1, astfel încît realizarea (3.32) este de tip autotestabil.

Raționamentul se repetă pentru celulele următoare, pînă la celula C_{n-1} inclusiv; structura fiecărei celule se deduce din precedenta prin suprimarea ieșirii cu indicele j cel mai mic.

3°. *Cazul celulelor* C_i, $i \leqslant k$. Pentru acest tip de celulă semnalele de intrare nu vor fi niciodată în configurația $(Z_k, 0)$, deoarece această configurație apare atunci cînd la k intrări primare ale celulelor anterioare a apărut bitul 1; dar în situația analizată nu există k celule anterioare ($i \leqslant k$), ci cel mult $k-1$ celule. Configurațiile semnalelor de intrare în această celulă sînt:

$$(Z_j, 0) \text{ sau } (Z_j, 1), \forall j \in [0, k-1]. \quad (3.34)$$

Pornind de la diagrama de tranziție definită anterior (fig. 3.27) se întocmește diagrama Karnaugh pentru minimizarea funcției y^k_k (fig. 3.32*a*). În prezența semnalelor de intrare corecte semnalul de ieșire y^k_k are doar două tranziții, $1 \to 1$ sau $1 \to 0$, și va fi o funcție de y^{k-1}_{k-1} și x_k. Rezultă:

$$y^k_k = y^{k-1}_{k-1} + \bar{x}_k. \quad (3.35)$$

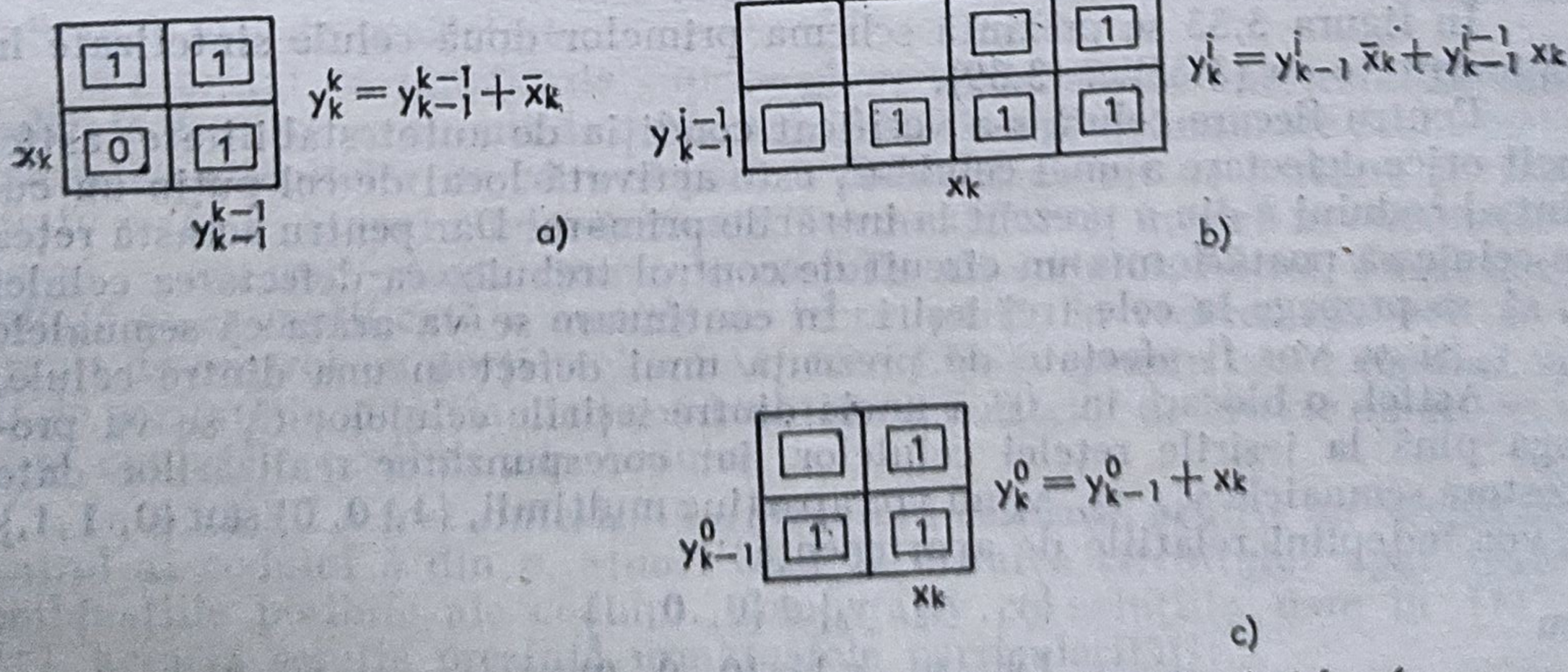


Fig. 3.32. Diagrame Karnaugh pentru sinteza celulelor C_i, $i \leqslant k-1$:
a — sinteza funcției y^k_k; *b* — sinteza funcției y^j_k $j \in (0, k)$; *c* — sinteza funcției y^0_k.

Vîrfurile active ale acestei funcții — în conformitate cu (3.34) — sînt: (0,0), (0,1), (1,0), (1,1), rezultînd că realizarea (3.35) este autotestabilă.

Pentru celula C_{k-1}, pe baza unui raționament analog, se deduce configurația semnalelor de intrare de forma:

$$(Z_j,\ 0) \text{ și } (Z_j,\ 1),\ \forall J \in [0,\ k-2]. \tag{3.36}$$

avînd structura identică cu (3.34). Rezultă că semnalul y_{k-1}^{k-1} este realizat de funcția logică dată de ecuația (3.35), la care indicele k este micșorat cu o unitate ș.a.m.d.

Semnalul de ieșire, y_k^j, $0 < j < k$, este o funcție de y_{k-1}^{j-1}, y_{k-1}^j și x_k conform celor arătate în graful tranzițiilor din figura 3.27 și matricea (3.23). Procedînd ca mai sus se întocmește diagrama Karnaugh (fig. 3.32*b*) pentru obținerea funcției y_k^j:

$$y_k^j = y_{k-1}^j\ \bar{x}_k + y_{k-1}^{j-1}\ x_k. \tag{3.37}$$

Analizîndu-se mulțimea vîrfurilor active, reprezentate prin pătrate în diagrama din figura 3.32*b*, se constată că realizarea (3.37) este autotestabilă, fiind îndeplinite condițiile de existență și dezvoltare ale teoremei 3.1.

Din graful tranzițiilor și matricea (3.23) se deduce că y_k^0 este o funcție de x_k și y_{k-1}^0 definită în diagrama Karnaugh din figura 3.32*c*, de unde rezultă:

$$y_k^0 = y_{k-1}^0 + x_k. \tag{3.38}$$

Realizarea dată de ecuația (3.38) este autotestabilă, deoarece vîrfurile active sînt (0,0), (0,1), (1,0) și (1,1) fiind îndeplinite astfel condițiile teoremei 3.1.

Soluțiile obținute pentru sinteza semnalelor y_k^k, y_k^j și y_k^0 cu ecuațiile (3.35), (3.37) și 3.38) sînt valabile pentru fiecare celulă C_i pînă la $i = 2$, modificîndu-se în mod corespunzător indicele k.

Celula C_1 are doar o intrare primară, x_1, și două ieșiri corespunzătoare lui y_1^0 și y_1^1, care codifică configurațiile Z_0 și Z_1 realizate de această celulă. În conformitate cu matricea (3.23) rezultă:

$$\text{și} \qquad \begin{aligned} y_1^0 &= x_1 \\ y_1^1 &= \bar{x}_1. \end{aligned} \tag{3.39}$$

În figura 3.33 se prezintă schema primelor două celule sintetizate în conformitate cu (3.35)... (3.39).

Pentru fiecare celulă s-a verificat condiția de autotestabilitate, astfel încît orice defectare a unei celule C_i este activată local de cel puțin un cuvînt al codului k din n prezent la intrările primare. Dar pentru această rețea de celule să poată forma un circuit de control trebuie ca defectarea celulei C_i să se propage la cele trei ieșiri. În continuare se va arăta că semnalele y_1, y_2 și y_3 vor fi afectate de prezența unui defect în una dintre celule.

Astfel, o blocare în „0" a uneia dintre ieșirile celulelor C_i se va propaga pînă la ieșirile rețelei celulelor, iar corespunzător realizărilor date acestora semnalele y_1, y_2, y_3 nu vor aparține mulțimii, $\{1,\ 0,\ 0\}$ sau $\{0,\ 1,1,\}$ ci vor îndeplini relațiile de apartenență:

$$\{y_1,\ y_2,\ y_3\} \in \{0,\ 0,\ 1\}$$

sau

$$\{y_1,\ y_2,\ y_3\} \in \{0,\ 0,\ 0\},$$

după cum x_n este „1" sau „0".

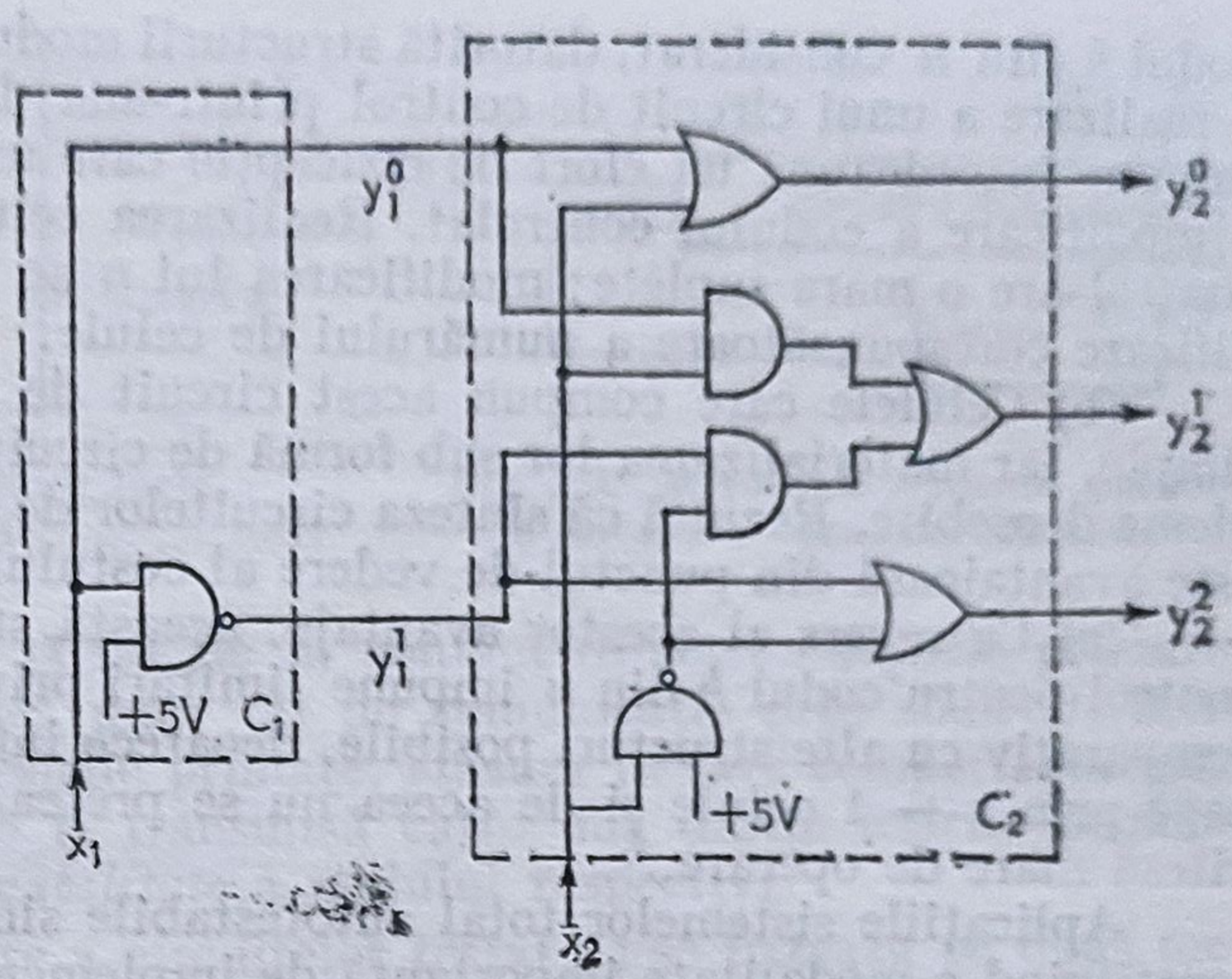


Fig. 3.33. Schema celulelor C_1 și C_2 ale circuitului de control autotestabil pentru codul 3 din 8.

Blocarea în „1" a uneia dintre ieșirile celulelor va determina ca toate ieșirile celulelor următoare să fie „1", ceea ce înseamnă că mulțimea ieșirilor celulelor vor fi în configurația Z_ε, care semnifică starea eronată; semnalele de la ieșirea lanțului de celule vor îndeplini condițiile:

$$\{y_1, y_2, y_3\} \in \{1,1,0\}$$

sau

$$\{y_1, y_2, y_3\} \in \{1,1,1,\}$$

după cum x_n este „1" sau „0", ceea ce conduce la o indicație de defectare în sistem.

3.3.4. CONCLUZII

Introducerea în § 3.3.1 — a unui model general de definire a sistemelor total autotestabile a permis prezentarea și analiza într-o manieră unitară a diferitelor structuri date acestor sisteme, ceea ce poate constitui un îndreptar util de proiectare.

S-a arătat că funcția de supraveghere a sistemelor autotestabile este realizată de către un circuit de control. În § 3.3.3. au fost analizate problemele concepției circuitelor de control pentru codurile cu bit de paritate și, respectiv *k* din *n*, atunci cînd combinațiile semnalelor de intrare în circuit formează — sau nu — toate cuvintele posibile ale codului. Astfel, a fost dezvoltată și exemplificată o metodă de lucru, utilă, în vederea sintezei unui circuit de control cu porți logice SAU EXCLUSIV pentru cazul codului cu bit de paritate în situația cînd combinațiile semnalelor de intrare nu acoperă toate cuvintele posibile ale codului.

În § 3.3.3.2 a fost sintetizată o structură celulară pentru un circuit de control al codului *k* din *n*, atunci cînd la intrarea circuitului apar toate combinațiile posibile ale codului. Comparativ cu soluțiile date în [11], [31], această soluție prezintă următoarele particularități:

(a) Investiția inițială de concepție odată efectuată, o realizare a unui circuit de control conform metodei propuse este foarte simplă oricare ar fi

codul k din n considerat, datorită structurii modulare adoptate. Din contră, o realizare a unui circuit de control printr-una din metodele [11], [31] va impune întotdeauna un efort de concepţie care nu este de loc neglijabil la o modificare a codului controlat. Realizarea celulară dată unui circuit de control are o mare supleţe; modificarea lui n se traduce doar printr-o modificare corespunzătoare a numărului de celule.

(b) Celulele care compun acest circuit de control realizează funcţii simple, iar materializarea lor sub formă de circuite integrate nu ridică probleme deosebite. Rezultă că sinteza circuitelor de control cu o reţea celulară este avantajoasă din punctul de vedere al costului acestor circuite.

(c) Ca revers al acestor avantaje, această structură a unui circuit de control pentru codul k din n impune limitări privind viteza de funcţionare comparativ cu alte structuri posibile, deoarece informaţia trebuie să se propage prin $n - 1$ celule şi de aceea nu se pretează la aplicaţii ce necesită viteză mare de operare.

Aplicaţiile sistemelor total autotestabile sînt numeroase, autotestarea constituind o modalitate importantă de implementare a toleranţei la defectări. Trebuie subliniat că pentru sistemele autotestabile nu creşte valoarea funcţiei de fiabilitate, din contră aceasta este mai mică, în schimb cresc credibilitatea şi securitatea acestor sisteme (capitolul 5) ,indicatori foarte importanţi pentru sistemele de mare răspundere funcţională.

Dintre realizările semnificative în domeniul sistemelor autotestabile se pot cita automate de tip sincron, procesoare şi microcalculatoare. În ceea ce priveşte sistemele secvenţiale de tip asincron, problemele concepţiei structurilor autotestabile sînt particulare şi antrenează o serie de dificultăţi caracteristice fiecărui sistem în parte [23], [31].

3.4. METODE DE ASIGURARE A UNEI TESTABILITĂŢI FACILE

Testabilitatea unui sistem constituie aptitudinea acestuia de detectare şi localizare uşoară a defectărilor posibile. Aceasta înseamnă că sistemul posedă un set de teste, uşor de generat, aplicat şi evaluat, care permite obţinerea unei localizări precise a defectărilor. Testabilitatea se realizează în faza de proiectare a sistemelor. Regulile de proiectare ce vizează testabilitatea încearcă, în general, să crească controlabilitatea şi observabilitatea sistemelor.

Analiza următoare va avea în vedere circuitele electronice.

S-a arătat că metoda generală de a produce programe de test pentru un circuit complex fig. (3.34) constă în a exersa funcţional fiecare bloc identificabil în circuit. Cum nu întotdeauna blocul cercetat are intrările accesibile în mod direct, stabilirea unor configuraţii de semnale pentru testare pe intrările sale presupune posibilitatea controlării valorilor logice ale nodurilor de reţea din regiunea A prin intermediul vectorilor de test aplicaţi pe intrările primare. Măsura în care este posibilă asigurarea de valori logice bine definite unui nod al reţelei, prin intermediul vectorilor aplicaţi pe intrările primare, poartă denumirea de *controlabilitate* a nodului respectiv.

Presupunîndu-se rezolvată problema configuraţiilor logice necesare pentru exersarea blocului logic testat, apare aspectul observării ieşirilor

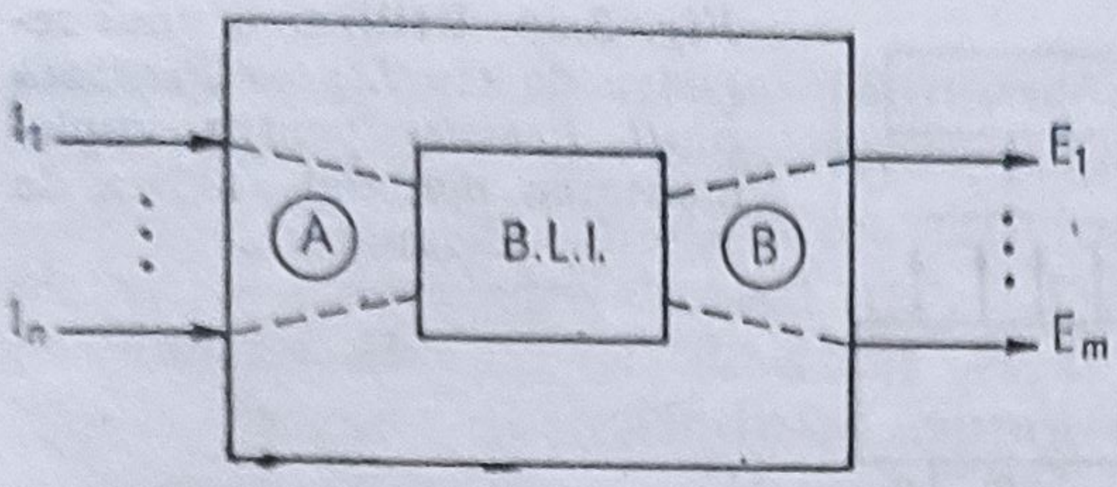


Fig. 3.34. Reprezentarea unui circuit logic complex de tip integrat.

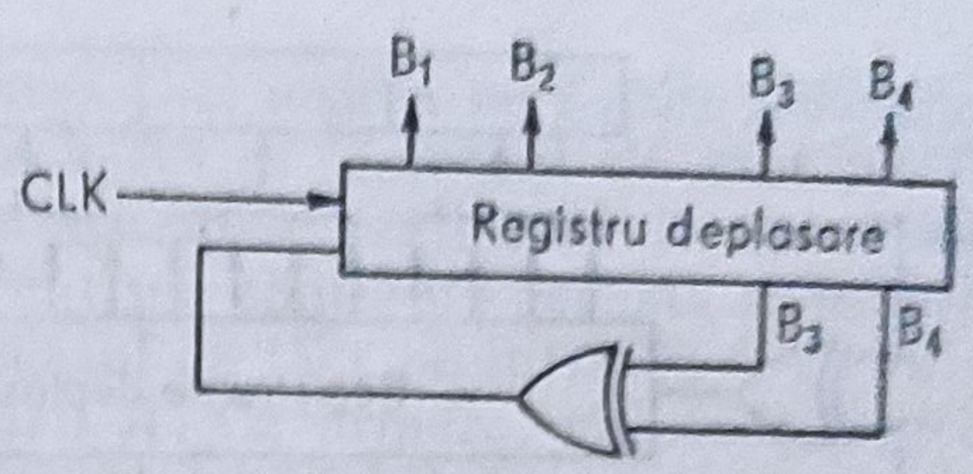


Fig. 3.35. Metodă de generare pseudo-aleatoare a vectorilor de test.

acestuia, deoarece ele pot să nu fie în contact direct cu ieșirile primare. Aceasta presupune stabilirea unor căi de comunicație prin zona *B*, între ieșirile blocului testat și ieșirile primare. Măsura în care comportarea unui nod al rețelei logice poate fi transmisă către una dintre ieșirile primare poartă denumirea de *observabilitate* a nodului respectiv.

Tehnicile de proiectare a circuitelor testabile se bazează pe două metode. Prima metodă pornește de la proiectarea existentă și introduce elemente logice adiționale; acestea pot conduce creșterea complexității circuitului, în termenii numărului de porți și numărului de interconexiuni, dar se reduce numărul de teste cerute. A doua metodă constă în proiectarea circuitelor, astfel încît ele însele să fie ușor testabile.

Aceste metode au în mod curent cîteva neajunsuri. În primul caz numărul porților și interconexiunilor poate deveni prohibitiv, iar a doua metodă poate introduce [49] întîrzieri mari în propagarea semnalelor și o creștere a numărului porților logice.

Prima metodă este ușor de aplicat în majoritatea cazurilor pentru rezolvarea problemelor de testabilitate la nivel plăci de circuite imprimate echipate cu componente și sisteme prin introducerea unor puncte de test suplimentare. În cazul circuitelor integrate, unde numărul terminalelor capsulei impune limitări drastice, nu se pot introduce puncte de test suplimentare decît cel mult în faza de fabricație a acestora, însă și la acest nivel apar limitări. În cazul circuitelor integrate de tip LSI sau VLSI implementarea primei metode se poate realiza prin introducerea unor posibilități de generare în circuit a unei secvențe de test pseudoaleatoare. Această generare a vectorilor de test poate fi obținută prin utilizarea unui registru de deplasare cu bucle de reacție (fig. 3.35) ce extrag informația din diferite puncte ale registrului și o transmit la intrarea acestuia (LFSR — *L*inear *F*eedback *S*hift *R*egister). Această metodă simplă, ușor de implementat în structura circuitului, permite obținerea prin metode nedeterministe a unui număr mare de vectori de test, $d_e = 2^{n-1}$.

În acest caz metodele de analiză a rezultatelor aplicării secvenței de test constau fie în numărarea tranzițiilor, fie în analiza de semnătură [32]. În metoda analizei de semnătură elementul constructiv fundamental este un LFSR, căruia i se aplică la intrare, printr-o poartă SAU EXCLUSIV două surse de informație: șirul de m biți rezultați în urma aplicării secvenței de test și șirul de semnale de pe bucla de reacție realizată cu prize de semnal în diferite puncte ale registrului (fig. 3.36). La terminarea secvenței de test, LFSR conține un cuvînt de n biți care poartă denumirea de *semnătură* a șirului de m biți prelucrați.

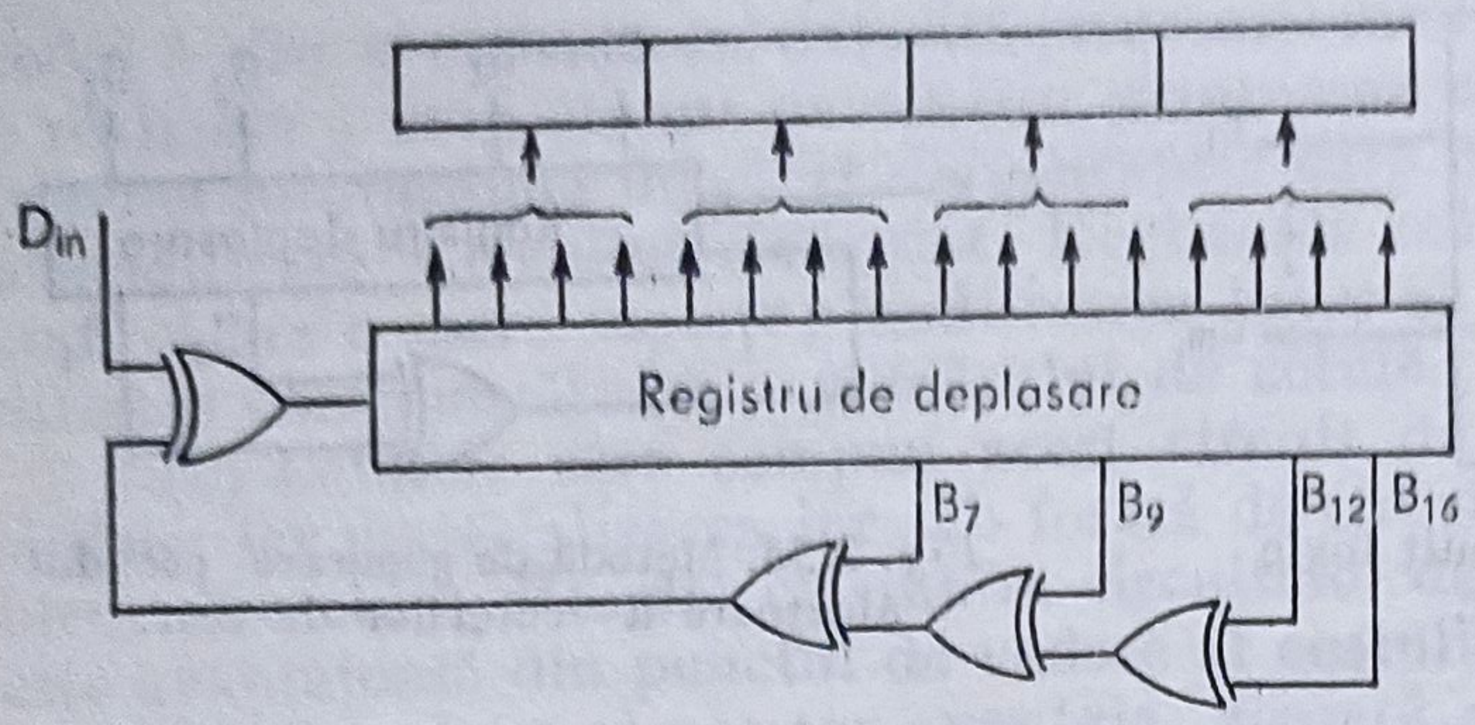


Fig. 3.36. Utilizarea unui registru de tip *Linear Feedback Shift Register* pentru implementarea metodei analiza de semnătură.

Cea de-a doua metodă presupune utilizarea — îndeosebi pentru circuitele electronice integrate de tip LSI sau VLSI — a tehnicii de proiectare cunoscută sub numele de LSSD (*L*evel — *S*ensitive *S*can *D*esign) [44], care se bazează pe următoarele două concepte:

(1) Circuitul trebuie să funcționeze sincron. Toate schimbările de stare ale circuitului sînt determinate de nivelul logic al semnalului de tact, nu de fronturile acestuia. Mai mult, stările stabile apărute ca răspuns la modificările semnalelor pe intrări sînt independente de întîrzierile datorate propagărilor pe căile de transfer (porți, interconexiuni etc.). Răspunsul este independent de ordinea în care se produc schimbările semnalelor de intrare (în cadrul unei aceleiași perioade de tact). Aceasta este proprietatea de sensibilitate la nivel menită să reducă dependența circuitului de parametrii săi de curent alternativ; degradarea fronturilor, întîrzieri de propagare etc.

(2) Circuitul trebuie să posede căi de acces serial. Acest deziderat se obține prin utilizarea unei structuri logice special proiectată în acest scop (fig. 3.37*a*), cunoscută sub denumirea SRL (polarity - Hold *S*hift *R*egister *L*atch). În figura 3.37*a*, B_1 și B_2 reprezintă două circuite bistabile interconectate în configurație master-slave. B_1 constituie dispozitivul de memorare a stării sistemului în funcționare normală, avînd intrarea de date pe

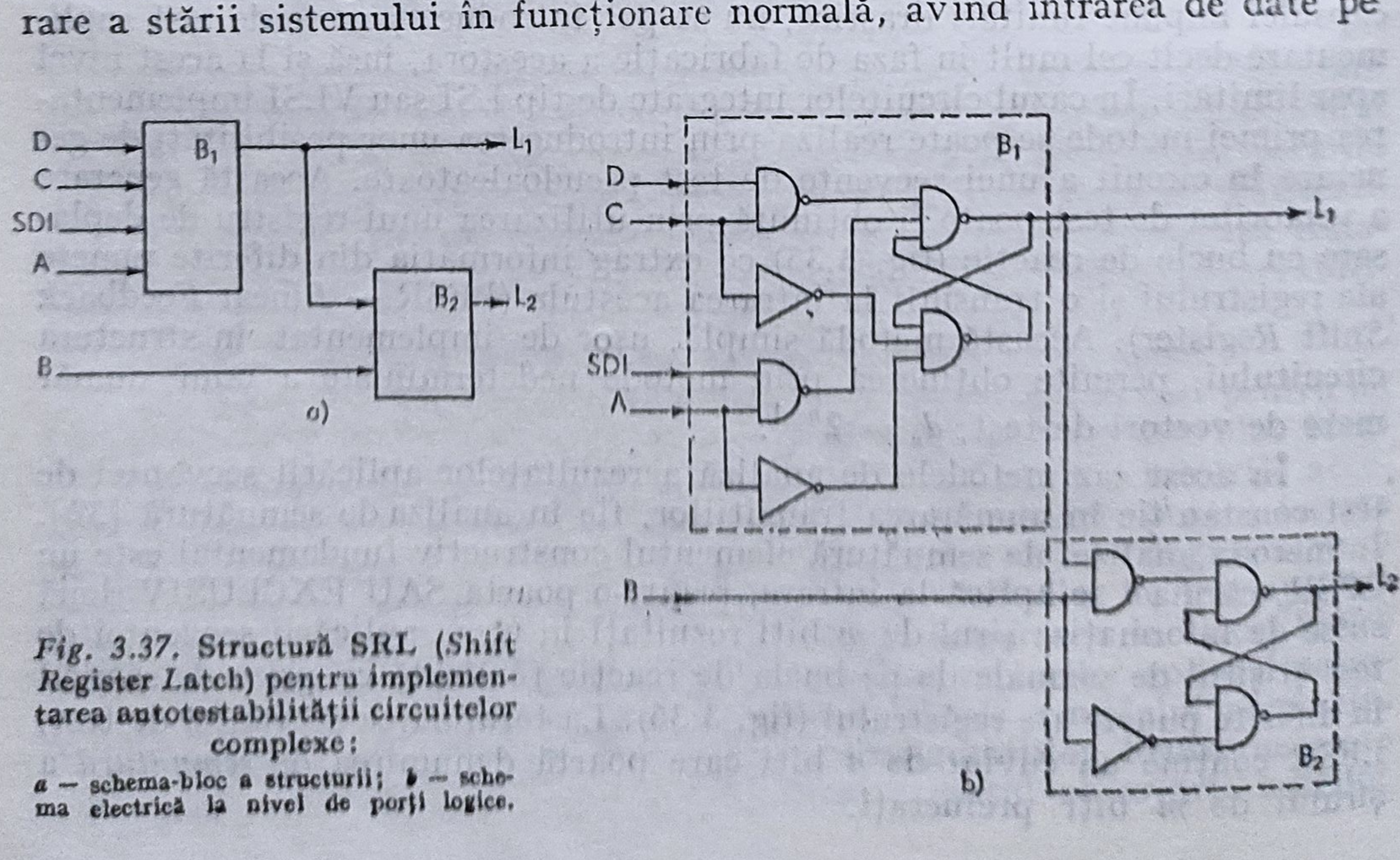


Fig. 3.37. Structură SRL (*S*hift *R*egister *L*atch) pentru implementarea autotestabilității circuitelor complexe:
a — schema-bloc a structurii; *b* — schema electrică la nivel de porți logice.

D, semnalul de tact al sistemului pe intrarea C și ieșirea pe L_1. Pe perioada funcționării normale semnalele „Scan Clock A" și semnalul de tact pe B sînt fixate în „O". Memorarea datelor din sistem se realizează la revenirea în „O" a semnalului de tact (fig. 3.37*b*). Pentru a configura această structură ca element al unei căi de acces serial semnalul de tact pe intrarea C trebuie blocat în „O", acționîndu-se asupra intrării A. Ca urmare, valoarea logică existentă pe intrarea SDI se încarcă în circuitul bistabil L_1, la revenirea în „O" a intrării A. Transferarea acestei informații — în continuare — în bistabilul L_2 se realizează la revenire în „O" a semnalului de tact aplicat pe intrarea B.

Realizarea unei testări automate în tehnica LSSD presupune următoarele etape:

(a) se testează registrele tip SRL (fig. 3.38), transformîndu-se în mod serial o secvență logică cunoscută;

(b) se testează blocurile combinaționale, transferîndu-se la intrarea lor vectorii de test necesari prin intermediul unui registru de deplasare — — *scan path*;

(c) se citește vectorul semnalelor rezultate la ieșirea blocului combinațional, extrăgîndu-se informația în mod serial prin intermediul registrului de deplasare;

(d) se repetă acțiunile descrise la punctele *b* și *c*, pînă la epuizarea programului de test.

Principalele avantaje oferite de tehnica de proiectare LSSD sînt:

— eliminarea dependenței bunei funcționări a sistemului de variabile dinamice (timpi de creștere și cădere a nivelurilor de semnal, timpi de propagare etc.);

— posibilitatea de a întrerupe bucla de reacție — prin trecerea circuitului în regim *scan-path* — eliminînîndu-se o cauză majoră a dificultăților întîmpinate în activitatea de diagnoză pentru circuitele logice secvențiale.

Ca dezavantaje a acestei tehnici se pot cita:

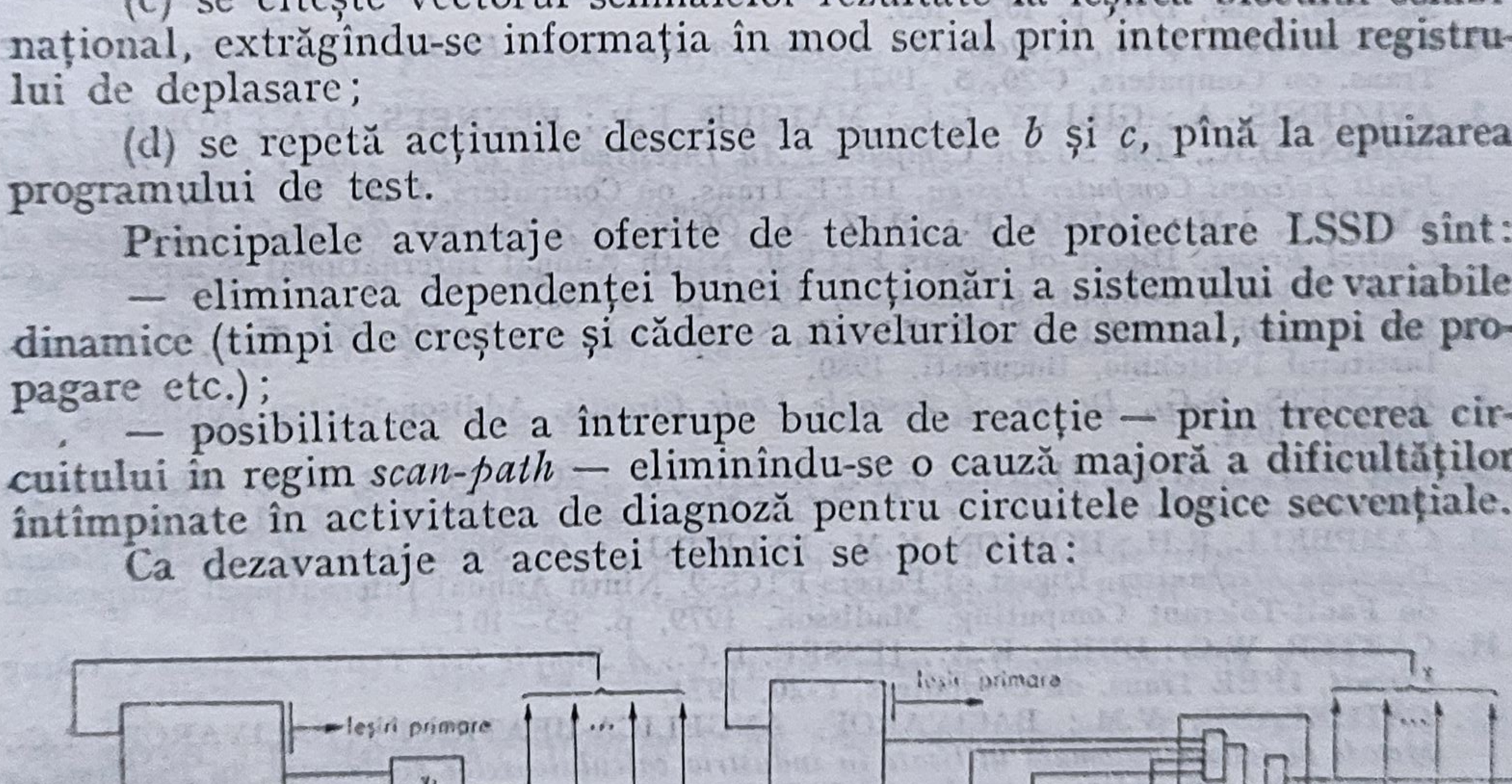


Fig. 3.38. **Utilizarea tehnicii LSSD (*Level-Sensitive Scan Design*) pentru implementarea autotestabilității**

a — descompunerea unui circuit complex în vederea testării automate; *b* — utilizarea structurilor SRL.

— creşterea numărului de porţi logice necesare implementării funcţiei logice, în varianta LSSD faţă de varianta clasică;

— obligaţia de a folosi o logică sincronă care impune o serie de restricţii de proiectare;

— imposibilitatea de a executa teste la viteza maximă a circuitului investigat, deci de a pune în evidenţă defecte dinamice.

Această tehnică de proiectare permite creşterea observabilităţii şi controlabilităţii circuitelor integrate de tip LSI, ceea ce atrage utilizarea acestor structuri în sistemele de înaltă fiabilitate — în general — şi în cele de tip tolerant la defectări, în special.

BIBLIOGRAFIE

1. ABAZI, Z.; THEVENOD ,P., *Test alèatoire de cartes à microprocesseur*, Rapport LAAS. N 87115, 1987.
2. ANDERSON, D.; METZE, G., *Design of Totally Self-Checking Check Circuits for* m *out of* n *Codes*, IEEE Trans. on Computers, **C-22**, 3, 1973.
3. ANDREWS, D.M., *Using Executable Assertions for Testing and Fault Tolerance*, Digest of Papers FTCS-9; Ninth Annual International Symposium on Fault Tolerant Computing, Madison, June, 1979, p. 102—105.
4. ARMSTRONG, D.B., *A Deductive Method for Simulating Faults in Logic Circuits*, IEEE Trans. on Computers, **C-20**, **5**, 1971.
5. AVIZIENIS, A.; GILLEY, G.C.; MATHUR, F.P.; RENNELS, D.A.; ROHR, J.A.; RODIN, D.K., *The STAR Computer : An Investigation of the Theory and Practice of Fault Tolerant Computer Design*, IEEE Trans. on Computers, **C-20**, **11**, 1971.
6. AYACHE, J.M.; AZEMA, P.; DIAZ, M., *Observer : A Concept for On-Line Detection of Control Errors*, Digest of Papers FTCS-9, Ninth Annual International Symposium on Fault Tolerant Computing, Madison, 1979, p. 79—86.
7. BACIVAROF, ANGELICA-BEATRICE, *Sisteme tolerante la defectări*, Teză de doctorat, Institutul Politehnic, Bucureşti, 1980.
8. BENNETS, R.G., *Design of Testable Logic Circuits*, Addison-Wesley Publishing Co, London, 1984.
9. BOURICIUS, W.G.; ILSIEN, E.P.; ROTH, J.P.; SCHNEIDER, P.R., *Algorithm for Detection of Faults in Logic Circuits*, IEEE Trans. on Computers, **C-20**, **11**, 1971.
10. CAMPBELL, R.H.; HORTON, K.M.; BELFORD, G.C., *Simulations of a Fault-Tolerant Deadline Mechanism*, Digest of Papers FTCS-9, Ninth Annual International Symposium on Fault-Tolerant Computing, Madisson, 1979, p. 95—101.
11. CARTER, W.C.; DUKE, K.A.; JESSEP, D.C., *A Simple Self Testing Decoder Checking Circuit*, IEEE Trans. on Computers, **C-20**, 1971.
12. CĂTUNEANU, V.M.; BACIVAROF, ANGELICA-BEATRICE; BACIVAROF, I.C., *Metode de testare automată utilizate în industria calculatoarelor electronice*, în *Probleme de automatizări*, **10**, Editura Academiei R.S. România, 1978, p. 45—52.
13. CĂTUNEANU, V.M.; BACIVAROF, ANGELICA-BEATRICE, *Echipament de testare automată a componentelor electronice*, Metrologia aplicată, **1**, 1978, p. 27—32.
14. CHANDRAMOULI, R., *On Testing Stuck-Open Faults*, Digest of Papers, FTCS, 13-th Annual International Symposium on Fault-Tolerant Computing, Milano, 1983, p. 258—265.
15. CHANG, H.Y.; MANNING, E.G.; METZE, G., *Fault Diagnosis of Digital Systems*. W.I., New-York, 1971.
16. CHUNG, F.R.K.; LEIGHTON, F.T.; ROSENBERG, A.L., *DICGENES : A Methodology for Designing Fault-Tolerant VLSI Processor Arrays*, Digest of Papers, FTCS 13-th Annual International Symposium Fault Tolerant Computing, Milano, 1983, p. 26—33.
17. CLANCY, C., *Qualitative Reasoning in Electronic Fault Diagnosis*, Electron Eng., **59**, **11**, 1987, p. 141—145.
18. DENT, J.J., *Diagnostic Engineering Requirements*, Proc. AFIPS, 1978, p. 503—507.
19. EDWARDS, C.R., *The Design of Easily Tested Circuits Using Mapping and Spectral Techniques*, The Radio and Electronic Engineer, **47**, **7**, 1977, p. 321—342.
20. FRIEDMAN, A.D., *Diagnosis of short circuit faults in combinational circuits*, IEEE Trans on Computers, **C-23**, **7**, 1974.

21. FUENTES, A.; DAVID, R.; COURTOIS, B., *Random Testing Versus Deterministic Testing of RAMS*, Digest of Papers FTCS, 16-th Annual International Symposium on Fault Tolerant Computing Systems, Vienna, Austria, 1986, p. 266–272.
22. FUJIWARARA, H.; SHIMONO, T., *On the Acceleration of Test Generation Algorithms*, Digest of Papers FTCS 13-th Annual International Symposium, Milano, 1983, p. 98–105.
23. GEFFROY, J.C., *Test en ligne et sécurité des systèmes logiques*, l'Université Paul Sabatier, Toulouse, France, 1976.
24. GIRARD, E.; RAULT, J.C.; TULLONE, R., *Les méthodes probabilietes de generation des sequences de test ; estimation de l'eficacité et de la longueur des séquences*, Rev. techn. Thomson CSF, 1, 1974.
25. GIRARD, E.; RAULT, J.C.; TULLONE, R., *La simulation des circuits logiques : problemes et mise en veuvre*, Rev. tech. Thomson CSF, 1, 1974.
26. GOLAN, P.; NOVAK, O.; HLAVICKA, J., *Pseudoexhaustive Test Pattern Generator with Enhanced Fault Coverage*, Digest of Papers FTCS, 16-th Annual International Symposium on Fault Tolerant Computing Systems, Vienna, Austria, 1986, p. 266–272.
27. KATSUKI, D. et all., *PLURIBUS – An Operational Fault-Tolerant Multiprocessor*, Proceedings of the IEEE, 66, oct., 1978, p. 1 146–1 159.
28. KU, C.T.; MASSON, G.M., *The Boolean Difference and Multiple Fault Analysis*, IEEE Trans. on Computers, C-24, 1, 1975.
29. LEBRUN, J.; ROBACH, CH.; SAUCIER, G., *Processor Testability and Design Consequences*, IEEE Trans. on Computers, 6, 1976.
30. LEVENDEL, J.; MENNON, P.R., *Fault Simulation*, Digest of Papers FTCS, 16-th Annual International Symposium on Fault Tolerant Computing Systems, Vienna, Austria, 1986, p. 278–286.
31. MAROUF, M.A.; FREEDMAN, A.D., *Efficient Design of Self Checking Checkers for m o: t of n Codes*, IEEE Trans. on Computers, C-27, 6, 1978.
32. MIDDLETON, T., *Functional Test Vector Generation for Digital LSI/VLSI Devices*, Proceedings of International Test Conference, Philadelphia, 1983, p. 682–691.
33. MOALLA, M., *L'Approche fonctionnelle dans la verification des systèmes informatique*, l'Institut National Polytechnique de Grenoble, France, 1976.
34. OHTSUKA, N. et. all., *A4-Mbit CMOS EPROM*, IEEE. J. Solid-State Circuits, SC-22, 5, 1987, p. 669–675.
35. PARHOMENKO, P.P. (redaktor), *Osnovî tehniceskoi diagnostiki*, Izd. Energia, Moskva, 1976.
36. PETERSON, W.W.; WELDON, E.J., *Error-Corecting Codes*, MIT Press, Cambridge, 1972.
37. RAULT, J.C., *Bibliographie sur le détection et la localisation des défauts dans les circuits logique*, Rapport Thompson CSF, 1975.
38. RAULT, J.C., *La détection et la localisation des défauts dans les circuits logiques*, Digest of Papers FTC-5, The Annual International Symposium on Fault Tolerant Computing, Paris, 1975.
39. ROBINSON, J.P., *Segmented Testing*, IEEE Trans. on Computers, C-34, 5 1985, p. 467–471.
40. ROTH, J.P., *Hardware Verification*, IEEE Trans. on Computers, C-26, 12, 1977, p. 1 292–1 294.
41. ROTH, J.P., *Computer Logic, Testing and Verification*, Computer Science Press, Potomac, Maryland, 1980.
42. SZIGENDA, S.A.; THOMPSON, E.W., *Modelling and Digital Simulation for Design Verification and Diagnostic*, IEEE Trans. on Computers, C-25, 12, 1976, p. 1 242–1 253.
43. TOY, W.N., *Fault-Tolerant Design of Local ESS Processors*, Proceedings of the IEEE, 66, 1978, p. 1 126–1 145.
44. VOICU, G., *Metode şi tehnologii de investigare a circuitelor integrate pe scară mare*. Referat de doctórat, Institutul Politehnic, Bucureşti, 1987.
45. WAKERLY, J.F., *Partially Self-Checking Circuits and Their Use in Performing Logical Operations*, IEEE Trans. on Computers, C-23, 7, 1974.
46. WILLIAMS, T.W., *Design for Testability, A Survey*, IEEE Trans. on Computers, C-31, 1, 1982, p. 2–15.
47. YANG, S.S.; YANG, S.C., *Multiple Fault Detection for Combinational Logic Circuits*, IEEE Trans. on Computers, C-24, 3, 1975.
48. * * * *Special Issues on Fault Tolerant Computers*, IEEE Trans. on Computers, 1971–1987.
49. * * * *Automatică, Management, Calculatoare*, 45, Editura Tehnică, Bucureşti, 1984.

CAPITOLUL 4

ANALIZA UNOR TEHNICI DE IMPLEMENTARE A TOLERANȚEI LA DEFECTĂRI

4.1. CONSIDERAȚII CRITICE ASUPRA TEHNICILOR DE IMPLEMENTARE A TOLERANȚEI LA DEFECTĂRI

În § 1.5 s-au reliefat cele două abordări de bază adoptate în implementarea toleranței la defectări, identificate în tehnicile de redondanță statică, respectiv dinamică, aplicate la nivelul hardware sau/și software al sistemului. În cele de urmează se prezintă o serie de particularități identificate la unele modalități de implementare a acestor tehnici.

Tehnicile de redondanță statică, aplicate la nivelul hardware al sistemului, se referă la utilizarea componentelor suplimentare care formează o parte permanentă a sistemului și servesc la mascarea semnalelor eronate generate de prezența unor defectări.

În schimb, un sistem cu structură redondantă de tip dinamic tolerează defectările prin reorganizarea activă a sistemului la apariția acestora, astfel încît o componentă defectă este efectiv înlocuită de o componentă în stare de bună funcționare. Această operație este realizată în trei etape, așa cum se reliefează în diagrama din figura 4.1. În prima etapă se realizează detecția defectării sistemului. Procedeele de test folosite urmăresc să detecteze defectul și să-l izoleze la o unitate înlocuibilă sau reparabilă. În etapa a doua defectul este îndepărtat prin repararea sau înlocuirea modulului defect cu modulul de rezervă ori prin reconfigurarea logică a sistemului în jurul modulului respectiv. În etapa a treia se realizează revenirea sistemului la o stare funcțională parcursă de acesta înainte de manifestarea defectului, după care operarea sistemului este reluată din acel punct.

În capitolele anterioare au fost analizate aspecte referitoare la primele două etape, urmînd ca problemele de bază ale celei de-a treia etapă să fie studiate în cadrul acestui capitol.

În general, orice procedeu de restabilire a funcționării unui sistem fie că este implementat printr-o structură redondantă dinamică sau nu constă — în mod uzual — din următoarele etape: detecția unui defect/unei erori, recunoașterea defectului/erorii, restabilirea funcționării sistemului și diagnosticarea defectului/erorii.

De cele mai multe ori, detecția defectului/erorii este o funcție realizată integral de o varietate de procedee implementate cu mijloace hardware.

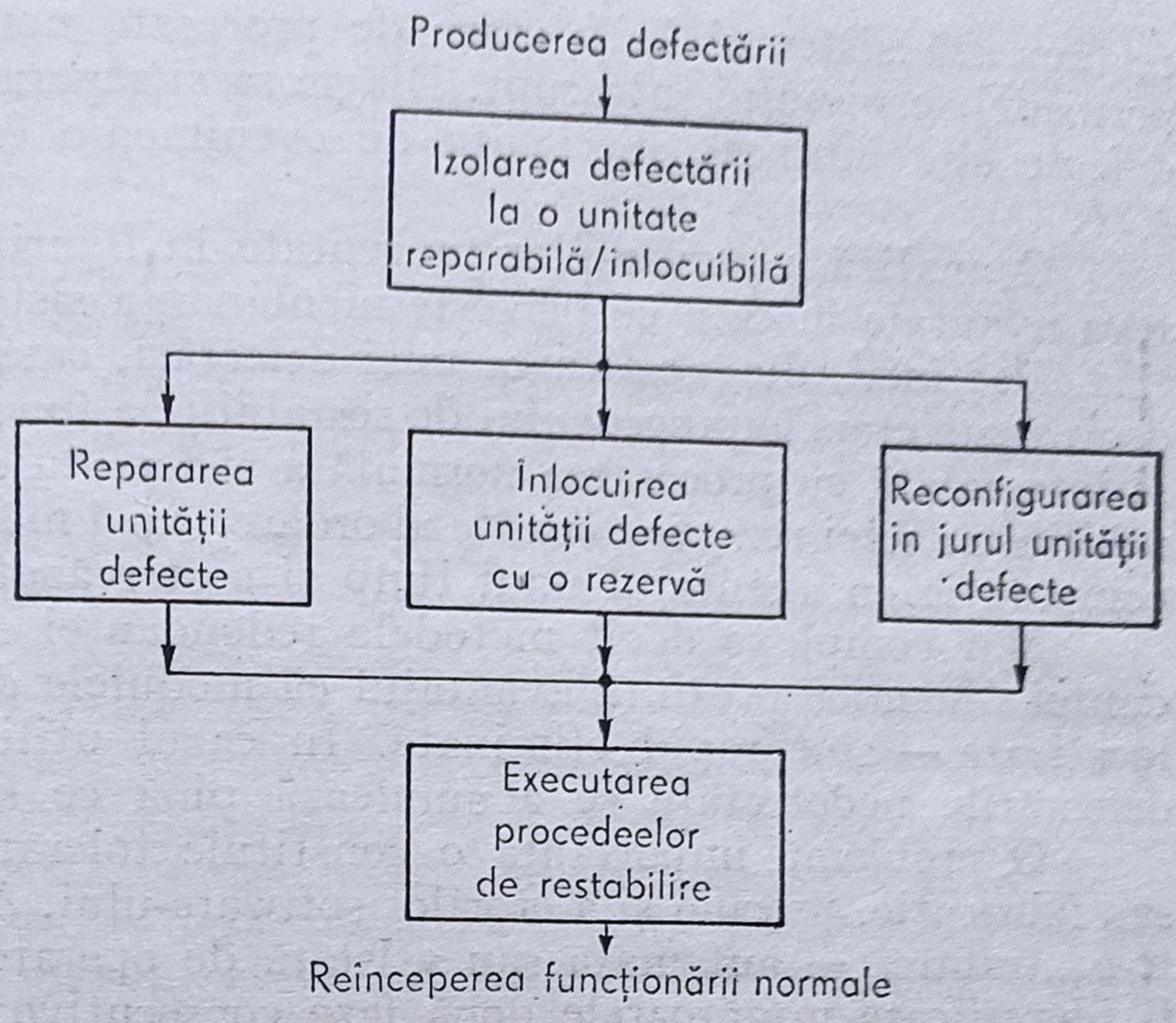


Fig. 4.1. Procedeul tolerării dinamice a defectărilor.

În toate cazurile testele de detecție sînt examinate de programe care asigură validitatea lor.

Obiectivul recunoașterii defectului/erorii constă în identificarea modulului funcțional la nivelul căruia are loc defectul/eroarea, precum și în deosebirea defectelor hardware permanente de cele tranzitorii.

Desfășurarea pasului următor are în vedere (in)disponibilitatea unei rezerve a modulului funcțional identificat.

Astfel, dacă nu este disponibilă o rezervă a modulului funcțional, atunci se inițiază acțiunile de diagnosticare. Operarea normală a sistemului, întreruptă în momentul detecției, este suspendată pentru diagnoză și reparare. În continuare sistemul trebuie restabilit la o anumită stare și într-un anumit punct din program de unde procesarea normală poate fi reluată. Secvența evenimentelor în acest caz este prezentată în figura 4.2*a*.

Atunci cînd este disponibilă o rezervă se adoptă o strategie diferită. Mai întîi sistemul este reconfigurat prin înlocuirea modulului defect cu rezerva corespunzătoare, utilizîndu-se o metodă de comutație automată, așa cum s-a arătat mai înainte. Apoi se inițiază un procedeu de restabilire, care

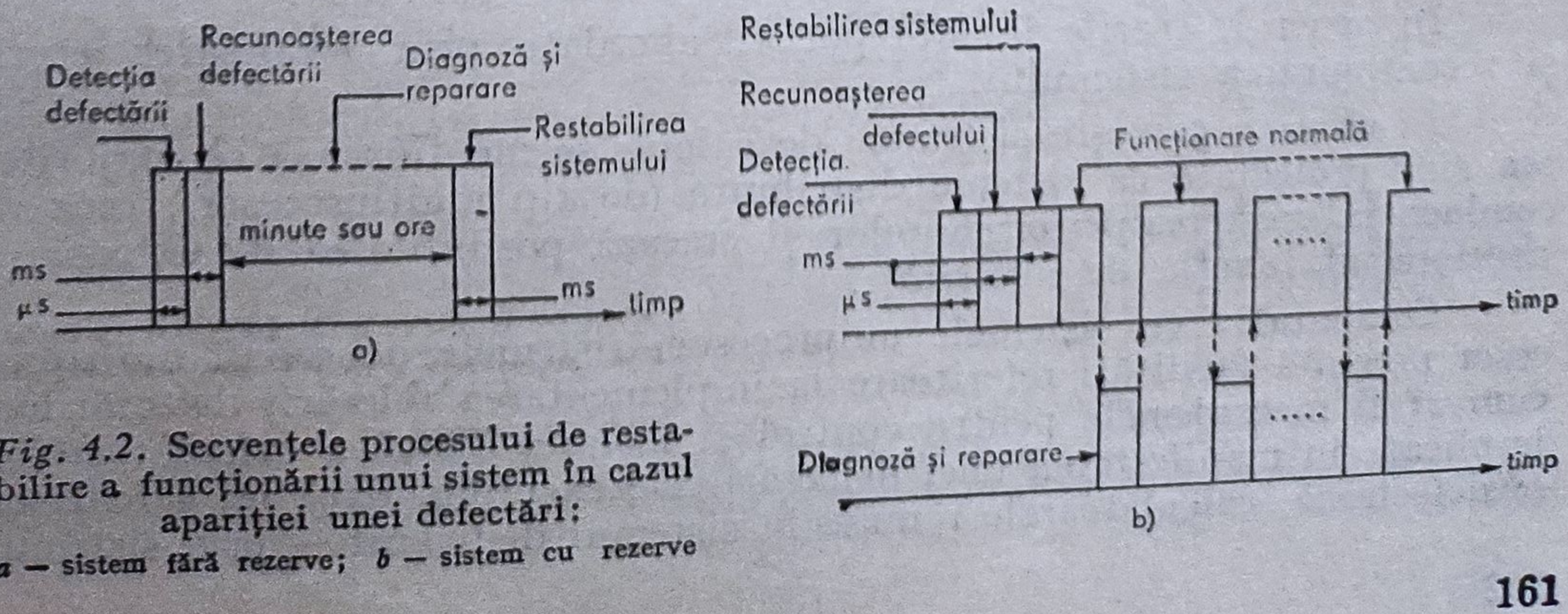


Fig. 4.2. Secvențele procesului de restabilire a funcționării unui sistem în cazul apariției unei defectări:

a — sistem fără rezerve; *b* — sistem cu rezerve

restaurează sistemul la o stare de procesare normală în scopul reducerii perioadei serviciului întrerupt. Diagnoza și repararea modulelor funcționale defecte sînt amînate. Secvența de evenimente este identificată în figura 4.2*b*.

O analiză a secvențelor prezentate în figurile 4.2*a* și *b* ilustrează cîteva avantaje în ceea ce privește mentenanța sistemelor tolerante la defectări. Mai întîi, diagnosticarea unei defectări, care consumă mai mult timp decît toate etapele procedeului de restabilire a funcționării, poate fi amînată și intercalată cu procesarea normală a sistemului. În al doilea rînd, disponibilitatea unei rezerve permite abordarea unei metode de diagnosticare prin comparație cu avantajul unui timp și cost scăzut.

Mai complexe decît metodele redondanței statice, procedeele redondanței dinamice prezintă avantajul că modulele defecte — după ce au fost localizate — sînt rapid eliminate. În cazul utilizării redondanței statice defectările nedetectate se acumulează pînă ce sistemul „cade" complet.

O problemă importantă o constituie tolerarea erorilor de proiectare sau fabricație, precum și a erorilor software-ului. Aceasta se realizează printr-o „tratare" — automată sau asistată de operator — a acestor erori. Pot fi identificate următoarele două faze consecutive: tratarea *erorilor efective* prin transformarea acestora în *erori latente* înainte ca sistemul să se defecteze și, respectiv, tratarea erorilor latente, destinată să prevină o nerevenire imediată a acestora.

Prima fază se poate realiza în două moduri: prin transformarea stării eronate a sistemului într-o stare presupusă lipsită de erori (*error recovery*), respectiv prin mascarea erorilor datorită informației redondante disponibile.

Procedeul *error recovery* poate adopta două forme:

- restabilirea funcționării sistemului prin readucerea sa într-o stare prealabilă stării eronate (*backward recovery*). Această stare — lipsită de eroare — în care rămîne sistemul constituie punctul de reluare a funcționării sale;
- corectarea anticipată a erorii (*forward recovery*), cînd transformarea stării eronate constă în a găsi o nouă stare, lipsită de erori, care n-a intervenit niciodată sau n-a fost ocupată de sistem de la ultima apariție a stării sale eronate.

Dacă se estimează că tratarea erorilor efective a avut capacitatea de a elimina direct eroarea, sau că șansele revenirii acesteia sînt suficient de reduse, atunci tratarea erorii latente nu se mai realizează.

În general, erorile latente pot fi abordate prin localizarea erorii și reconfigurarea sistemului.

În ceea ce privește sistemele de calcul, se menționează că majoritatea sînt proiectate ca sisteme distribuite (de tip multiprocesor), ceea ce conduce la creșterea performanțelor și creează posibilitatea de extensie a acestora în funcție de necesități.

Comparativ cu sistemele uniprocesor, sistemele de calcul multiprocesor prezintă facilități referitoare la implementarea tolerării defectărilor, cum ar fi mecanismele pentru control și izolarea fizică a calculatoarelor implicate în manifestarea unei defectări. De exemplu, dacă un calculator se defectează, calculatoarele rămase în sistemul distribuit — prin sistemul

software disponibil — detectează erorile și reconfigurează corespunzător sistemul. Într-un sistem uniprocesor detecția și restabilirea se realizează cu mijloace hardware, motivul constînd în aceea că este inoportun ca astfel de operații să fie executate de software-ul calculatorului defect.

De subliniat faptul că în cazul implementării toleranței la defectări pentru sistemele multiprocesor apar probleme suplimentare comparativ cu sistemele uniprocesor. Astfel, procesoarele componente introduc întîrzieri variabile în propagarea diferitelor mesaje, diferențe care nu pot fi neglijabile.

În consecință, pentru procesele care cooperează într-un sistem distribuit trebuie introdus un sistem de control specific. În acest mod, schimbările de stare într-un proces — importante pentru alte procese — sînt transmise corespunzător în vederea realizării unor decizii compatibile. Sincronizarea este dificil de obținut în prezența diferitelor întîrzieri. De exemplu, dacă un procesor este deconectat datorită identificării unei defectări, celelalte procesoare din sistem vor „ști" aceasta numai după ce mesajul de întrerupere va fi recepționat de fiecare dintre ele. În timpul tranziției mesajului respectiv celelalte procesoare pot trimite mesaje către calculatorul deja deconectat.

O altă problemă este aceea că sistemele distribuite sînt probabil sisteme eterogene, datorită diversității cerințelor funcționale și diversității alimentărilor. Codurile diferite adoptate ca și principiile de programare a procesoarelor vor complica detecția defectărilor, ca și restabilirea sistemului. Este important ca mecanismele tolerării defectărilor în sistemele distribuite să fie flexibile și selective, deoarece procesoarele componente vor avea grade diferite de tolerare a defectărilor, iar cerințele de fiabilitate se schimbă în timpul vieții sistemului. O abordare posibilă pentru astfel de sisteme o constituie proiectarea de tip *layering* [25].

4.2. TEHNICI DE RECONFIGURARE A SISTEMELOR LA APARIȚIA DEFECTĂRILOR

Tehnicile de reconfigurare a sistemelor au rolul de a asigura localizarea elementelor defecte, eliminarea sau izolarea acestora și introducerea automată a unor rezerve. În general, aceste tehnici sînt implementate pe cale hardware.

O cerință fundamentală a tehnicilor de reconfigurare se referă la protecția sistemului față de propagarea influenței unității defecte precum și la stabilirea frontierelor logice și fizice de propagare a erorilor.

4.2.1. PARTICULARITĂȚI ALE TEHNICILOR DE LOCALIZARE A DEFECTĂRILOR ÎN CAZUL UNOR SISTEME TOLERANTE LA DEFECTĂRI

Procedeul cel mai utilizat în acest scop este cel al testelor de diagnoză. După cum s-a specificat în capitolul 3, un test de diagnoză implică introducerea unor seturi de semnale la intrările primare pentru care sînt cunoscute

semnalele corecte la ieșirile primare. O neconcordanță în vectorul semnalelor de ieșire evidențiază prezența unui anume defect în sistem. Testele de diagnoză se utilizează de obicei pentru localizarea defectelor de tip hardware, dar ele pot fi folosite și în sistemele software, ca un mijloc de a indica dacă un program este eronat sau nu.

Se menționează utilizarea pe scară largă a testelor de diagnoză în cazul sistemului de comutație electronică tolerant la defectări ESS (*Electronic Switching System*) No 1A [20] (fig. 4.3). Detecția inițială a unei defectări se realizează prin identificarea unei neconcordanțe între funcțiile celor două procesoare, activ și în rezervă. Urmează apoi localizarea defectării într-o zonă mai restrînsă, realizată în două etape. În prima etapă un test simplu de diagnoză identifică care C.P.U. este defectă, după care aceasta este comutată în afara modulului de procesare normal, protejînd sistemul de propagarea efectului unei defectări. Apoi se procedează la reluarea unor programe de diagnoză, care izolează defectul la cea mai mică componentă înlocuibilă. Odată ce defectul a fost localizat este trimis un mesaj operatorului uman, care înlocuiește modulul specificat de programul de diagnoză cu un modul de rezervă.

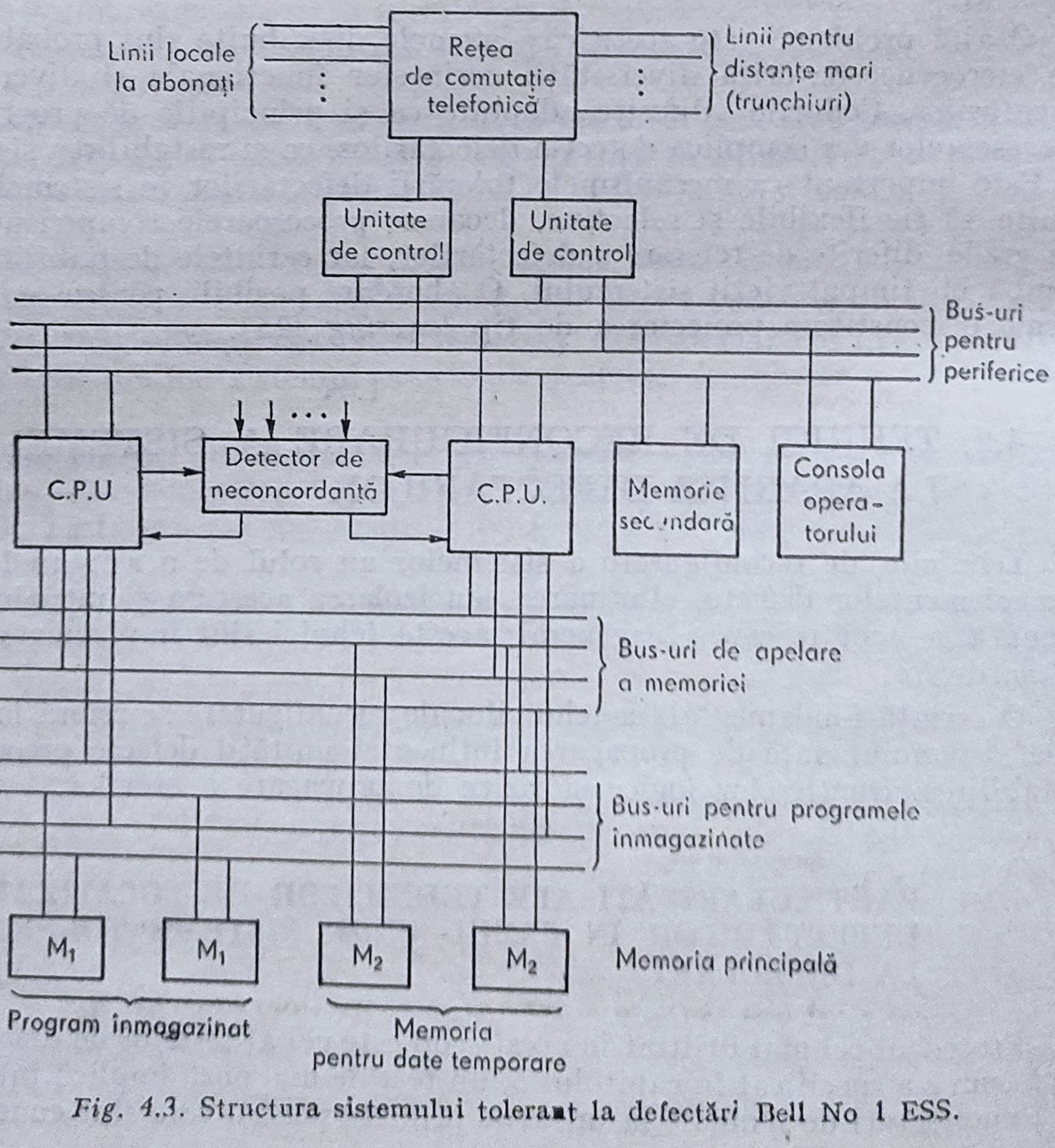


Fig. 4.3. Structura sistemului tolerant la defectări Bell No 1 ESS.

În general, nu se recomandă [2] utilizarea extensivă a testelor de diagnoză în cazul sistemului software, datorită timpului cerut pentru execuția lor deoarece implică o întrerupere substanțială în funcționarea normală a sistemului.

Pentru sistemele tolerante la defectări PLURIBUS sau TANDEM se remarcă o altă abordare [35], [47], acestea fiind proiectate astfel încît este permisă scoaterea unor componente din funcționarea curentă pentru aplicarea testelor de diagnoză. În cazul în care aceste componente sînt în stare de bună funcționare este posibilă reintegrarea lor în sistemul operant. Această operație poate fi realizată — în general — atunci cînd sistemul funcționează *on line* în scopul detectării defectărilor latente.

Pentru a reduce costul realizării testelor de diagnostic în sistemele cu structură redondantă se pot utiliza comparațiile între semnalele de ieșire ale diferitelor componente din structura sistemului care realizează aceleași funcții. Rezultatele obținute de un element al sistemului suspectat de a fi defect sînt comparate cu rezultatele elementului identic din altă copie a sistemului considerat în bună stare de funcționare. Existența unei eventuale neconcordanțe indică prezența unui defect.

În cazul sistemului tolerant la defectări FTMP (*F*ault *T*olerant *M*ulti *P*rocessor) [32] prezența unui defect este indicată de un astfel de test realizat de către voterele sistemului. Localizarea defectului se bazează pe realocarea adecvată a componentelor la diferite bus-uri. Cînd la unul din votere apare o neconcordanță între valorile semnalelor recepționate, toate componentele care pot transmite conflictual în triadă sînt alocate la bus-uri diferite, astfel încît componentele sau bus-urile defecte să poată fi identificate repede, dacă defectele sînt de tip permanent. Urmînd un procedeu asemănător pot fi ținute sub control și defectările tranzitorii ale componentelor sau bus-urilor, urmînd ca la un număr de defectări tranzitorii apriori stabilit, respectivele componente să fie scoase din serviciul activ. Cu un test final, o componentă „suspectă" poate fi temporar scoasă din funcțiune pentru a observa dacă aceasta conduce la dispariția problemelor.

4.2.2. PARTICULARITĂȚI ALE TEHNICILOR DE REPARARE AUTOMATĂ A SISTEMULUI

În urma etapei de localizare a defectărilor, una sau mai multe componente ale sistemului vor fi considerate ca defecte. Pentru a preveni ca aceste componente suspecte să influențeze viitoarea funcționare normală a sistemului, trebuie avute în vedere măsuri de reparare.

În cele ce urmează se consideră că repararea sistemului nu implică executarea unor operații de „reparare" propriu-zisă a acestor componente, ci numai a unor acțiuni de reconfigurare a sistemului într-un mod care să conducă la modificarea gradului de utilizare a componentelor suspecte de defectare. Tehnicile de reconfigurare a sistemului pot fi clasificate în următoarele categorii [7]:

— *manuale*, cînd toate acțiunile sînt realizate de factori externi sistemului, uzual factori umani;

— *dinamice*, cînd acțiunile de reconfigurare sînt realizate de către sistem, dar numai ca răspuns la instrucțiuni din afara sistemului;

— *spontane*, cînd toate acțiunile sînt inițiate și realizate de către sistem însuși.

Datorită prețului mai ridicat, tehnicile de reconfigurare dinamice și spontane sînt întîlnite în cazul sistemelor tolerante la defectări pentru care tehnicile manuale sînt fie inaccesibile — de exemplu, sistemul JPL STAR — sau la cele pentru care întîrzierile introduse de acestea nu sînt acceptabile — de pildă, sistemele tolerante la defectări SIFT, FTMP ș.a. [39]

Reconfigurarea manuală constă în modificarea manuală a conexiunilor între componente sau substituirea pe aceeași cale a componentelor defecte. Tehnicile de reconfigurare dinamice și spontane încearcă să automatizeze aceste acțiuni, prin utilizarea diferitelor forme de rețele de comutație care au ca scop să modifice interconexiunile între componente. În figura 4.4 este indicată o astfel de structură.

Comutatorul prezintă interfețe cu ambele componente de comutat. Comutația de la o componentă la alta modifică ansamblul de interacțiuni al sistemului, dar sistemul însuși rămîne neschimbat.

Atunci cînd reconfigurarea manuală este neadecvată sau nedisponibilă este mai indicat să se folosească o tehnică de reconfigurare de tip dinamic, aceasta și datorită faptului că pentru multe sisteme defectul se localizează manual și deci nu se mai pune problema unei reconfigurări spontane.

Cea mai simplă strategie de reconfigurare a sistemului — adoptată în general pentru reconfigurarea sistemelor hardware — este plasarea componentelor într-o configurație de tip *stand-by*" (v. capitolul 2). Dacă erorile de proiectare nu sînt considerate ca fiind esențiale, elementul de rezervă poate fi identic — din punctul de vedere constructiv și din cel al proiectării — cu elementul funcțional. Atunci cînd se urmărește înlăturarea erorilor de proiectare este necesar ca rezervele să fie proiectate în mod independent (deci diferit) de elementele de bază; este în special cazul tolerării erorilor software, iar metodele ce pot fi utilizate în această situație — cum sînt programarea N-versională și metoda blocului de restabilire — vor fi analizate mai detaliat în § 4.4.

Pentru multe sisteme tolerante la defectări (FTMP, PLURIBUS, PRIME sau SIFT), în scopul obținerii unui avantaj major din prezența rezervelor, se adoptă următoarea abordare: inițial toate componentele — funcționale și de rezervă — sînt disponibile să asigure servicii, iar dacă o componentă este suspectată de a fi defectă, atunci ea este înlăturată din sistem. În acest mod rezultă o capacitate superioară a sistemului de a

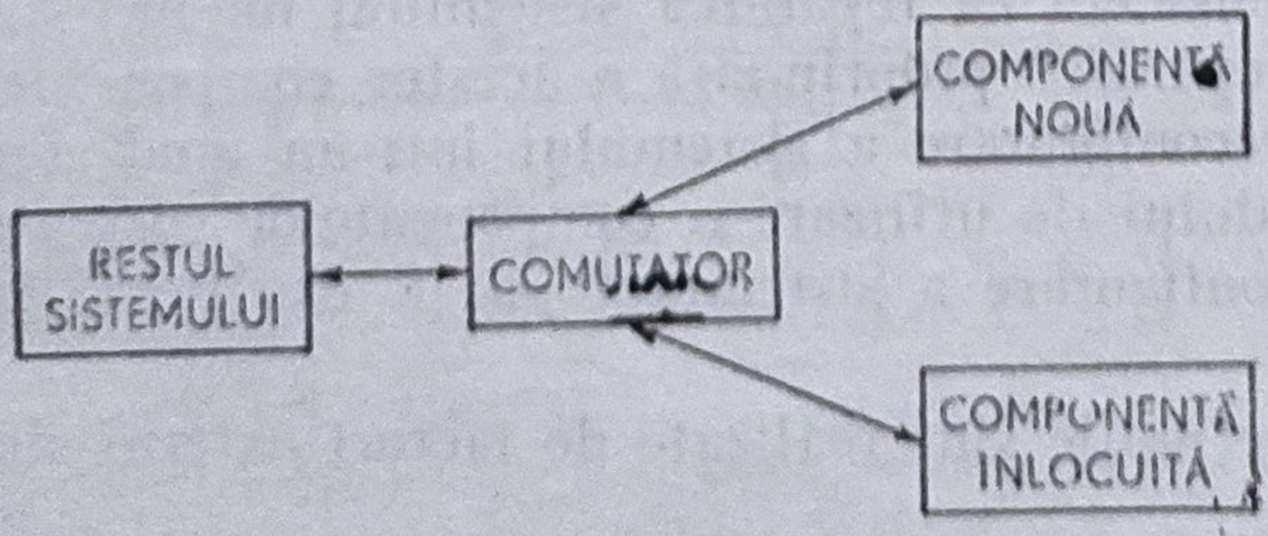


Fig. 4.4. Reconfigurarea structurii unui sistem prin comutație.

suporta suprasolicitări, precum și un grad de tolerare a defectărilor superior celui specificat inițial. Această operare va conduce la o degradare lentă (*graceful degradation*) a serviciilor asigurate de sistem, deoarece componentele rămase vor prelua sarcinile celor ce se vor defecta. De exemplu, în sistemul cu structură tolerantă la defectări pentru comanda navetei spațiale [2] setul inițial de 4 calculatoare identice poate fi redus de către echipaj la trei, două sau chiar un singur calculator, fără nici o pierdere a facilităților asigurate, efectul constînd doar în reducerea capacităților de tolerare a defectărilor. Un alt exemplu de structură de tip *graceful degradation* este memoria tolerantă la defectări propusă în [2]. În acest sistem un modul de memorie defect este izolat din setul de module operaționale, cu o reducere corespunzătoare a dimensiunilor memoriei principale.

Dimensiunile unităților înlocuibile influențează atît efortul necesar localizării defectelor, cît și costul întregului sistem și depind de mai mulți factori. În capitolul 2 s-au analizat detaliat aspectele privind identificarea dimensiunilor optime din punctul de vedere al cîștigului de fiabilitate adus și al costului necesar. Se poate constata că unitățile mari permit o localizare mai ușoară a defectării, dar impun în mod corespunzător o redondanță la nivel superior; de aici poate rezulta un cost mai ridicat al sistemului. Dimpotrivă, unitățile mici conțin mai puține componente și ca rezultat pot fi proiectate pentru a avea — cel puțin teoretic — o medie a timpului de bună funcționare de valoare mai mare; în schimb vor necesita o rețea de comutație mai complexă.

Un alt factor care influențează mărimea unităților înlocuibile îl constituie nivelul la care este posibilă reparația. De pildă, în interiorul unui microprocesor sau al modulului de memorie nu mai este necesară efectuarea testelor de diagnoză, deoarece în cazul menționat aceste subsisteme — nereparabile — sînt înlocuite integral (de exemplu, cazul modulelor de procesoare ale sistemului SIFT).

Analiza diferitelor sisteme operaționale cu structură tolerantă la defectări evidențiază și alte strategii de reconfigurare. Astfel, sistemul de comutație ESS No 1 A utilizează o structură de tip *stand-by* numai pentru acele componente identificate ca fiind susceptibile de defectare. Atunci cînd se încearcă o înlocuire neadecvată, sau dacă nu este disponibilă nici o rezervă, configurația de circuite a procesorului este activată cu scopul de a realiza o configurație minimală capabilă să efectueze testele de diagnoză a erorilor pentru toate componentele. Cele identificate ca aflîndu-se în stare de bună funcționare sînt adăugate la configurația minimală a sistemului pînă ce este restabilită funcționarea normală a acestuia. Strategiile de reconfigurare globală a sistemului ESS No 1 A oferă un exemplu de localizare a defectelor efectuat în combinație cu repararea sistemului.

Și în cazul altor sisteme tolerante la defectări, cum sînt sistemele PRIME și PLURIBUS, a fost adoptată strategia de a porni de la un sistem minimal de „încredere", care este adus la situația de sistem operațional prin încorporarea sistematică a componentelor ce satisfac testele de diagnoză. De exemplu, pentru sistemul PRIME [13] se pornește de la un procesor „sigur", completîndu-se treptat configurația de lucru după „trecerea" testelor de diagnoză. În cazul sistemului PLURIBUS reconfigurarea este realizată sub controlul unui sistem de operare, urmîndu-se o secvență de etaje prestabilite. În acest mod sînt adăugate treptat sistemului reconfi-

gurat — după ce sînt testate și găsite ca fiind corespunzătoare — un generator de tact, un procesor, o unitate de memorie, bus-uri și module de intrare-ieșire.

Reconfigurarea se realizează de un sistem de comutare adecvat. De menționat că în cazul sistemelor PLURIBUS și FTMP reconfigurarea sistemului software utilizează unele facilități ale comutației de tip hardware. În schimb, la sistemele SIFT și JPL STAR mecanismele de comutație sînt implementate atît hardware cît și software. La conceperea sistemului trebuie avut în vedere ca sistemul de comutație să nu fie activat în mod accidental. Astfel, accesul la comutatorii de reconfigurare ai sistemului PLURIBUS este controlat de un mecanism de tip *password*, asigurîndu-se securitatea funcționării sistemului.

De menționat că un sistem trebuie să dispună de facilități pentru o reparare manuală chiar și în condițiile în care el are posibilitatea unei reconfigurări dinamice sau spontane. Restaurarea integrală a capacității de tolerare a defectărilor presupune o utilizare optimală a resurselor disponibile, ceea ce poate fi realizat numai manual. Pentru sistemele de telecomunicații ESS No 1 A și TANDEM 16, care trebuie să asigure un serviciu de 24 ore din 24, ca și pentru alte sisteme de mare răspundere funcțională au fost prevăzute și posibilități de mentenanță manuală în paralel cu funcționarea *on-line*. Aceasta este realizată mai ușor pentru sistemele care funcționează duplex sau care au un grad de redondanță ridicat (cum sînt sistemele ESS, PLURIBUS și TANDEM).

O problemă specială a intervenit pentru unele dintre primele variante ale unor sisteme tolerante la defectări (PLURIBUS ș.a.): repararea manuală era mult îngreunată datorită lipsei de informații referitoare la defectări, mascarea excesivă conducînd la pierderea evidenței defectărilor sistemului. De aici a rezultat necesitatea — pentru generațiile următoare ale acestor sisteme — prevederii unei culegeri independente de date privind comportarea eronată a sistemului, avînd ca scop facilitatea mentenanței manuale a acestor sisteme.

4.3. MODEL AL RECONFIGURĂRII ÎN SISTEMELE MULTIPROCESOR

4.3.1. INTRODUCERE

Tehnicile de implementare a toleranței la defectări utilizate în sistemele uniprocesor pot fi aplicate și în cazul sistemelor multiprocesor. Apar însă diferențe în realizarea diagnosticării defectărilor și a restabilirii funcționării. În timp ce în sistemele multiprocesor aceste operații pot fi realizate de software-ul sistemelor rămase în funcțiune, în cele uniprocesor restabilirea automată a funcționării este controlată prin hardware, deoarece procesorul defect nu mai poate executa procesul de software pînă ce nu este reparat.

Așa cum s-a menționat, o modalitate de realizare a restabilirii funcționării sistemului la apariția unei defectări o constituie *reconfigurarea* acestuia. Pentru sistemele uniprocesor, reconfigurarea este realizată de o structură

redondantă de comutație (v. capitolul 2). În sistemele multiprocesor apariția unui defect la un procesor atrage totodată și întreruperea conexiunilor cu celelalte procesoare, iar restabilirea bunei funcționări înseamnă nu numai înlocuirea procesorului defect, ci și restabilirea conexiunilor acestuia cu celelalte procesoare din lanț.

În cele ce urmează, pornindu-se de la o analiză topologică a rețelei de procesoare și utilizîndu-se formalismul din teoria grafelor [29] se dezvoltă un model de implementare a reconfigurării unei rețele multiprocesor, care va fi optimală din punctul de vedere al costului, acceptînd un număr minim de rezerve pentru tolerarea unui număr dat de defecte.

Nodurile grafului reprezintă componentele hardware — în cazul studiat procesoare (calculatoare) —, iar arcele — liniile de comunicații între procesoarele componente, de exemplu rețele de comutație sau bus-uri. Un astfel de graf este un graf topologic și descrie sistemul din punct de vedere structural, în timp ce grafele care descriu modelele structurale de fiabilitate sînt grafe funcționale.

4.3.2. IPOTEZE ȘI DEFINIȚII

Se consideră un sistem multiprocesor care conține procesoare cu caracteristici identice. Prin urmare, nodurile grafului considerat pentru descrierea structurii sistemului sînt de același tip, iar arcele — nedirecționale. Apariția unui defect în sistem este reprezentată de înlăturarea nodurilor și arcelor din graf, corespunzătoare procesoarelor afectate.

Pentru dezvoltarea modelului se adoptă în continuare următoarele definiții:

Definiția 4.1. Sistemul în configurație minimală este sistemul care realizează performanțele cerute, dar nu tolerează nici un defect;

Definiția 4.2. Un graf de bază, G_b, este graful configurației minimale a sistemului.

Sistemului cu structură redondantă, care poate tolera un număr de defecte, i se atașează graful G_r, ce va conține în structura sa pe G_b ca un subgraf al său.

Definiția 4.3. Un graf G_r este redondant, dacă îl conține pe G_b ca un subgraf al său.

Deci, în structura lui G_r există la orice moment de timp un subgraf particular, G_b, care este izomorfic lui G_b, $G_b' = G_b$.

Graful G_r este astfel privit ca o realizare tolerantă la defectări a sistemului reprezentat de G_b, iar G_b' reprezintă un sistem activ angajat în procesarea datelor la un moment dat.

Partea rămasă din graful G_r, $G_r - G_b'$, reprezintă componentele neutilizate (rezervele) sau neutilizabile (defecte). Astfel, fiecare nod x al fiecărui graf G_r poate fi văzut ca avînd trei stări posibile:

(1) activă, $x \in G_b'$

(2) rezervă, (3) defect $\Big\}$ $x \in G_r - G_b'$

Se presupune că sistemul analizat conține un mecanism pentru autodiagnoză continuă. De exemplu, fiecare procesor poate fi testat de unul sau mai multe procesoare vecine. Odată ce este identificată o defectare a unui nod activ, se inițiază un proces de restabilire care realizează înlocuirea grafului G_b' cu un alt graf $G_b'' = G_b$, ce nu conține nici un nod defect. Aceasta înseamnă că, atunci cînd G_b' conține k noduri defecte, cel puțin k noduri în rezervă trebuie trecute în stare activă și incluse în G_b''. Modalitatea în care este determinat noul graf activ, G_b'', constituie strategia de restabilire adoptată. În cele ce urmează restabilirea este privită ca un proces de reconfigurare în jurul nodurilor defecte. Schimbările posibile în starea unui nod sînt ilustrate în figura 4.5.

În continuare se introduce conceptul de graf „k tolerant la defectări" [30], notat k-FT.

Definiția 4.4. Graful G_r este k-tolerant la defectări, dacă eliminarea oricăror k noduri și a arcelor conectate la aceste noduri conduc la un graf izomorfic lui G_b sau care îl conține pe G_b.

Procesul de restabilire implică o cantitate considerabilă de informație transferată de-a lungul arcelor grafului. De exemplu, un nod în rezervă, r, activat pentru înlocuirea unui nod defect, x, trebuie să primească toată informația, definind atît funcțiile lui x cît și starea acestuia la ultima testare cînd era în stare de bună funcționare. Informația este transferată lui r de către x sau de un alt nod care memorează starea lui x (de exemplu, un sistem supervizor). Numărul de noduri fără defecte, a căror stare sau identitate este schimbată atunci cînd se formează G''_b din G'_b, se apreciază ca o măsură a timpului de restabilire și conduce la următoarea definiție, necesară în dezvoltarea ulterioară a modelului.

Definiția 4.5. Graful G_b este posibil de restabilit prin t modificări din G_r, adică este de tip t-SR, dacă graful G_r este k-tolerant la defectări, și se poate restabili prin orice defect care afectează un număr de k noduri, $k \leqslant t$, prin schimbarea stării sau identității a t noduri fără defecte.

În multe cazuri restabilirea poate fi realizată prin înlocuirea a k noduri din G'_b cu k noduri în rezervă. Acestea sînt grafele de tip k-FT. Se consideră că nodurile în rezervă, atunci cînd sînt activate, se află în stare de bună funcționare. În anumite condiții, în procesul reconfigurării apare necesitatea de a înlocui nodurile active cu alte noduri active sau cu rezerve, ori ca un nod activ să-și schimbe identitatea cu un alt nod activ, de aici rezultînd condiția $k \leqslant t$ din definiția de mai sus. Parametrul t definit aici este dependent de strategia de restabilire utilizată, ca și de alegerea configurației active inițiale, G'_b.

Combinîndu-se definițiile 4.4 și 4.5, se poate enunța următoarea definiție:

Definiția 4.6. Graful G_r, avînd graful de bază G_b, este de tip k-FT/t-SR dacă G_r se poate restabili la apariția a k defecte în cel puțin t pași de restabilire, rezultați prin schimbarea stării a t noduri sau modificarea identității lor.

În general, $k \leqslant t$. Atunci cînd $k = t$, în conformitate cu definiția 4.5, graful este de tip t-SR.

E x e m p l u l 4.1. Se consideră un sistem de bază format din două procesoare, avînd graful caracteristic G_b (fig. 4.6 *a*) și sistemul reconfigurabil corespunzător reprezentat de graful G_r (fig. 4.6 *b*).

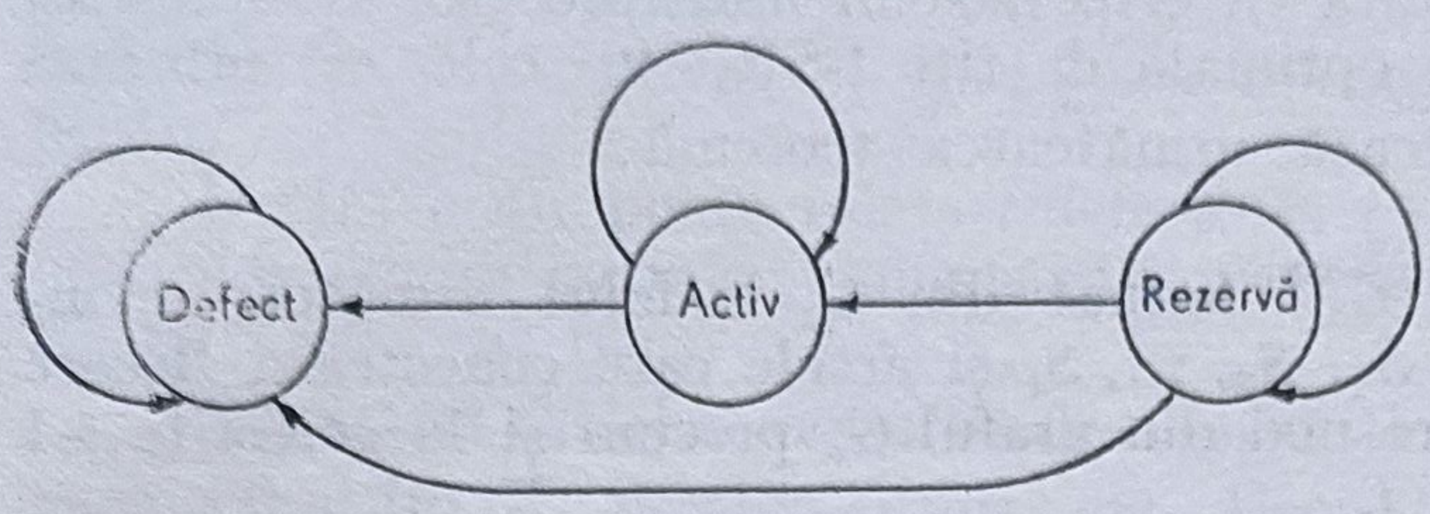


Fig. 4.5. Diagrama stărilor unui nod dintr-un graf al unui sistem.

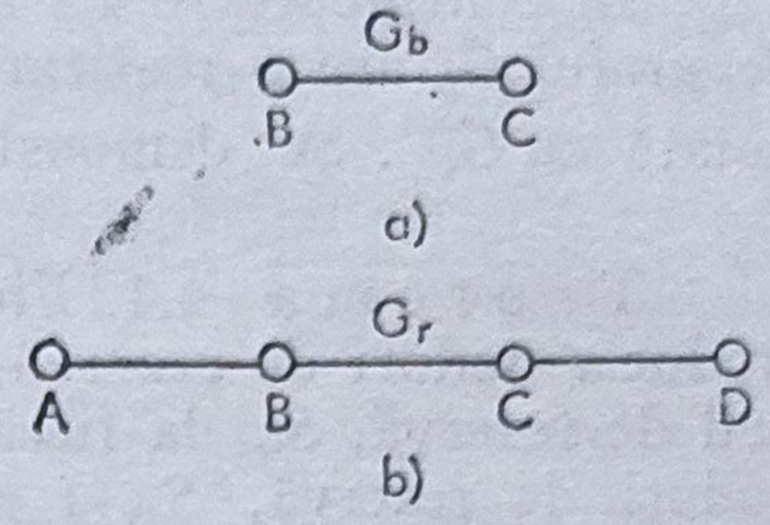


Fig. 4.6. Sistem care se poate restabili într-o etapă:
a — graful sistemului de bază;
b — graful sistemului redondant.

Defectarea nodului B sau a nodului C este permisă, deoarece G_r conține în structura sa grafele G'_b și G_b'' formate din nodurile C D, respectiv A B, grafe izomorfe cu G_b. Restabilirea se realizează într-un pas prin înlocuirea nodului B cu nodul D, iar atunci cînd nodul C este cel defect, prin înlocuirea acestuia cu nodul A. Nu este acceptată defectarea a două noduri pentru realizarea unei reconfigurări automate. În acest caz rezultă că G_r este un graf de tip 1-FT.

Dacă graful de bază este format din nodurile A și B, atunci restabilirea la defectarea nodului B se va realiza în două etape: A este replasat în locul nodurilor C sau D, iar D, respectiv C, va lua locul nodului defect, B; deci, în acest proces există două noduri care își schimbă starea, respectiv identitatea. În situația prezentată sistemul este de tip 2-SR.

Calculul mărimilor de restabilire k și t este dificil. Astfel, pentru a determina dacă G_r tolerează un defect dat, este necesar să se determine dacă graful G_r^F — reprezentînd graful sistemului defect — conține un subgraf izomorf lui G_b. Aceasta constituie o problemă legată de izomorfismul (sub) grafelor. Devine necesară examinarea tuturor subgrafelor conținute de G_r izomorfe lui G_b, în scopul de a determina dacă G_r este t-SR în raport cu G_b. De menționat că aspectele generale ale izomorfismului subgrafelor se caracterizează printr-o anumită complexitate din punctul de vedere al calculelor, dar că există algoritmi utili pentru anumite clase de grafe; în ceea ce privește cazul general se cunoaște un procedeu euristic eficient [30].

4.3.3. MODELUL RECONFIGURĂRII OPTIMALE ÎN t PAȘI

Este necesar ca grafele izomorfe G'_b și G''_b să fie astfel formate, încît restabilirea să se efectueze într-un timp cît mai mic. Timpul de restabilire este minim atunci cînd nici un nod activ în stare de bună funcționare din G'_b nu este afectat pentru formarea lui G''_b și numai nodurile de rezervă sînt utilizate pentru înlocuirea nodurilor defecte. Dezideratul menționat se realizează atunci cînd există t noduri de rezervă în G_r care înlocuiesc cele t noduri defecte din G'_b, rezultînd astfel graful G''_b izomorf cu acesta. Graful G_r acceptă astfel t defecte și se restabilește în t pași, adică este de tip t-SR. Mai mult, în aceste condiții graful G_r este optimal t-SR.

În continuare se va arăta că orice sistem multiprocesor caracterizat de graful G_b are o realizare optimală de tip t-SR, în cele ce urmează notată cu G_t^{opt}. Se demonstrează următoarea teoremă:

T e o r e m a 4.1. Fie G_t^{opt} format din G_b în felul următor: se introduc t noduri de rezervă, S_1, S_2, ..., S_t și arcele care conectează fiecare nod de rezervă, S_i, la fiecare nod din graful G_b precum și la celelalte t-1 noduri de rezervă, S_j, $i \neq j$.

Un graf redondant, G_r, al grafului de bază, G_b, este t-SR optimal, dacă și numai dacă $G_r = G_t^{opt}$.

D e m o n s t r a ț i e. Pentru început se va arăta că G_t^{opt} cu G_b graful sistemului original activ este de tip t-SR. Fie X un nod oarecare defect. Nodul X poate fi înlocuit într-un pas de restabilire de orice nod de rezervă, S_i, dacă S_i este adiacent tuturor nodurilor din graful G_b adiacente lui X. Orice secvență de t noduri defecte poate fi înlocuită în mod similar dacă G_t^{opt} include t noduri de rezervă față de G_b și fiecare rezervă înlocuiește nodul defect într-un pas. Astfel, t noduri de rezervă permit celor t noduri defecte din G_b să fie înlocuite în t pași, implicînd că G_t^{opt} este de tip t-SR în raport cu graful de bază, G_b.

Fie G_t^* un graf cu t noduri de rezervă de tip t-SR. În continuare se va arăta că G_t^* este o realizare optimală atunci cînd conține un subgraf izomorf cu G_t^{opt}, sau chiar $G_t^* = G_t^{opt}$. Fie G_b^* graful de bază — subsistemul activ — al lui G_t^*, astfel încît $G_b^* = G_b$. Nodurile lui $G_t^* - G_b^*$ sînt S_1^*, S_2^*, S_3^*,..., S_t^*.

Rămîne să se arate că fiecare nod S_i^*, $i = 1, 2, ..., t$, este adiacent nodurilor din graful G_t^*, caz în care $G_t^* = G_t^{opt}$.

Se presupune că S_i^* nu este adiacent nodurilor Y_j^*. Există două cazuri:

C a z u l 1. Nodul Y_j^* îndeplinește condiția $Y_j^* \in G_b^*$. Graful G_b^* conține cel puțin două noduri. Fie $Y_k^* \in G_b^*$; totodată să presupunem că Y_j^* și Y_k^* sînt adiacente.

Se consideră că o secvență de t noduri defecte afectează nodul Y_k^* și fiecare dintre cele $t - 1$ noduri de rezervă.

Rămîne doar nodul S_i^* care trebuie să fie utilizat pentru a înlocui nodul Y_k^*, deoarece G_t^* este de tip t-SR și există numai t noduri de rezervă disponibile, inclusiv nodul S_i^*. Dar, conform ipotezei de mai sus, S_i^* nu este adiacent nodului Y_j^*, iar nodurile Y_j^* și Y_k^* formează un arc al grafului G_b^*; prin urmare Y_k^* nu poate fi înlocuit de S_i^*. În consecință, G_t^* **nu e**ste de tip t-SR, ceea ce contrazice ipoteza inițială.

Deci, pentru ca graful G_t să fie de tip t-SR este necesar ca S_i^* să fie adiacent fiecărui nod al grafului G_b^*.

C a z u l 2. Se presupune că $Y_j^* \in G_t^* - G_b^*$; fie $Y_j^* = S_j^*$. Se consideră o secvență de t noduri defecte. La apariția celui de-al $t - 1$-lea defect mai există nodurile de rezervă S_i^* sau S_j^* care urmează să fie activate. Fie S_j^* nodul activat. Nodul S_j^* are cel puțin un nod vecin, Z_k^*, care aparține grafului activ curent. În continuare, se presupune că următorul defect este al nodului Z_k^*. Nodul S_i^* va fi nodul de rezervă disponibil să-l înlocuiască

pe Z_k^*. Dacă S_i^* nu este adiacent nodului S_j^*, care aparține acum sistemului activ, înseamnă că S_i^* nu-l poate înlocui pe Z_k^* într-o singură etapă pentru a reconfigura sistemul activ.

Deci, G_t^* nu poate tolera t defecte în t pași, apărînd astfel o contradicție cu ipoteza inițială și anume că G_t este de tip t-SR. Prin urmare, nodul S_i^* este adiacent fiecărui nod $S_j^* \neq S_i^*$.

S-a arătat astfel că nodurile de rezervă ale lui G_t^* sînt conectate la fiecare nod al lui G_b^*, precum și între ele. Deci, G_t^* și G_t^{opt} sînt izomorfe. Rezultă de aici că fiecare sistem t-SR optimal este izomorf cu G_t^{opt}.

E x e m p l u l 4.2. În figura 4.7.*a* este reprezentat graful unui sistem multiprocesor format din patru procesoare, iar în figura 4.7*b* — graful sistemului optimal de tip 2-SR, I_2^{opt}, obținut prin procedeul sugerat de teorema 4.1.

Defectarea oricăror două procesoare din sistemul redondant reprezentat de graful din figura 4.7*b* nu afectează buna funcționare a sistemului. Pe graf se remarcă ușor că defectarea oricăror secvențe de două noduri nu afectează identificarea unui graf $G_b' = G_b$.

Acest model de reconfigurare — considerat optimal — are totuși dezavantajul că gradul maxim al unui nod care aparține grafului G_t^{opt} poate fi foarte mare. Astfel, dacă G_b conține n noduri, atunci nodurile de rezervă în G_t^{opt} au gradul $n + t - 1$, acesta fiind gradul maxim al unui nod într-un graf de $n + t - 1$ noduri. Fizic, gradul unui nod corespunde *fan-out*-ului unui procesor sau numărului de porți intrare-ieșire; or, din considerente fizice, acestea sînt în număr limitat. În cazul microprocesoarelor, datorită

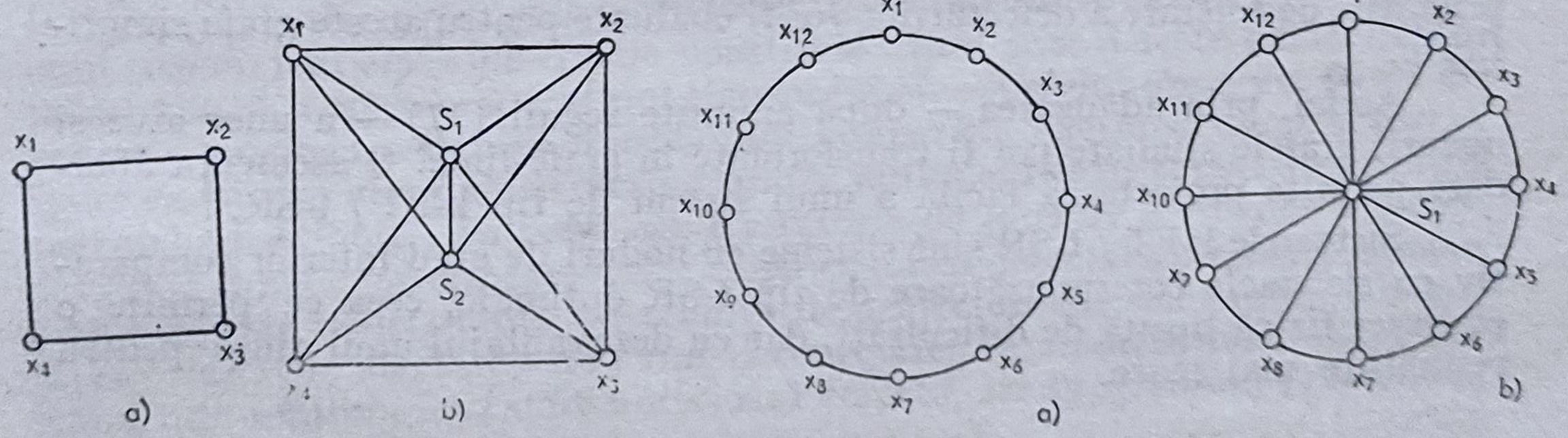


Fig. 4.7. Sistem de tip 2-SR optimal:
a — graful sistemului de bază; *b* — graful sistemului I_2^{opt}.

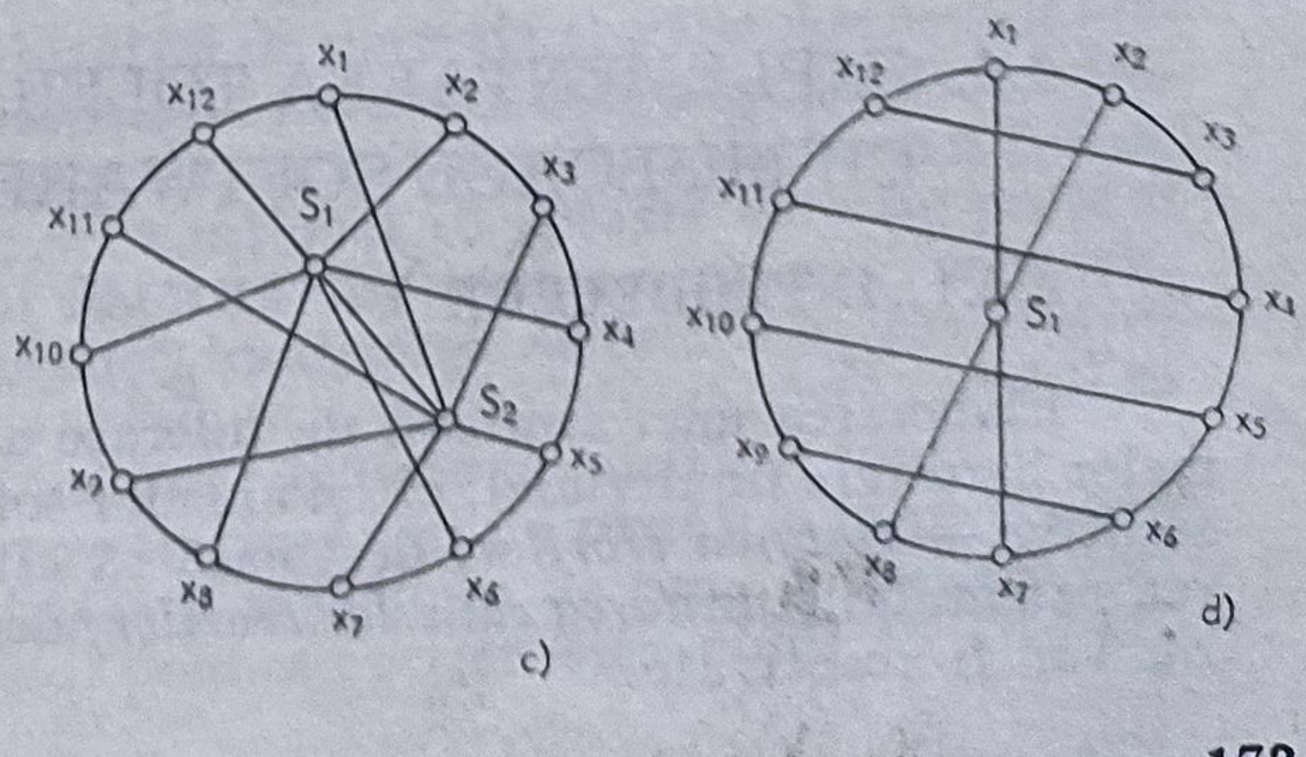


Fig. 4.8. Realizări tolerante la defectări de tip 1-FT/t-SR ale sistemului C_{12}:
a — graful sistemului neredondant C_{12}; *b* — graful sistemului 1-FT/1-SR optimal; *c* — graful unui sistem 1 — FT/1-SR neoptimal; *d* — graful unui sistem de tip 1-FT/9-SR.

numărului limitat de pini ai capsulei circuitului integrat respectiv, numărul căilor paralele de date este restrictiv. Deoarece gradul unui nod este limitat, pentru rezolvarea problemei este interesant de luat în considerare și sistemele reconfigurabile neoptimal, cu un număr de pași de restabilire diferit de numărul defectărilor tolerate.

E x e m p l u l 4.3. În figura 4.8 sînt date diferite realizări 1-FT ale grafului de bază C_{12}, un graf cu 12 noduri reprezentat în figura 4.8*a*. În figura 4.8*b* se prezintă realizarea 1-FT / 1-SR optimală a lui C_{12}, construită în conformitate cu teorema 4.1. Se observă că nodul de rezervă — nodul central — are gradul 12. În figura 4.8*c* este ilustrată o altă versiune de tip 1-FT / 1-SR a lui C_{12}, neoptimală cu două noduri în rezervă, S_1 și S_2. Gradul maxim al unui nod este în acest caz egal cu 7. Micșorarea numărului de arce incidente la un nod a atras introducerea unui nod suplimentar. Graful din figura 4.8*d* este o realizare 1-FT / 9-SR a lui C_{12}, care — așa cum este demonstrat în [29] — conține un număr minim de arce. Se observă că această realizare are gradul cel mai mic al nodurilor. Gradul maxim al unui nod este în acest caz egal cu 4.

În proiectarea unui astfel de sistem reconfigurabil apar de soluționat problemele privind numărul de rezerve, gradul maxim al unui nod și numărul pașilor de restabilire.

Construirea unui graf corespunzător pentru un sistem k-FT / t-SR este dificilă, ca de altfel și calculul parametrilor k și t atunci cînd se consideră un graf anume. Problema poate fi abordată în două moduri și anume [38]:

(1) Se pot considera o serie de proprietăți ale grafelor în vederea simplificării analizelor referitoare la posibilitățile de reconfigurare la defectarea nodurilor;

(2) Grafele date se transformă în grafe ușor de analizat, astfel că proprietățile de tolerare a defectărilor vor fi evaluate pentru aceste grafe simplificate.

Astfel, prin adăugarea — după anumite reguli [27] — a unor arce și noduri, grafele studiate pot fi transformate în grafe-linie. O asemenea abordare permite proiectarea facilă a unui sistem de tip k-FT / t-SR.

Sistemele k-FT / t-SR sînt sisteme cu noduri de grad inferior comparativ cu sistemele corespunzătoare de tip t-SR optimale, ceea ce permite o realizare fizică lipsită de dificultăți, dar cu dezavantajul unui timp pentru restabilire mai mare.

4.4. IMPLEMENTAREA TOLERANȚEI LA ERORI CU MIJLOACE SOFTWARE

4.4.1. INTRODUCERE

Elaborarea unei strategii de tolerare a defectărilor atît în cazul sistemelor hardware (materiale), cît și al celor software comportă două etape consecutive — *tratarea erorilor*, destinată să elimine erorile privind funcționarea sistemului, și *tratarea cauzelor erorilor*, destinată să asigure că aceste erori nu vor fi reactivate.

Analiza următoare va avea în vedere sistemele software.

Implementarea tratării erorilor poate fi realizată în două moduri:

(a) prin *acoperirea erorilor*, atunci cînd starea eronată a sistemului este mascată instantaneu sau este transformată într-o altă stare, presupusă fără erori; la rîndul său, această transformare, care presupune introducerea unor metode specifice de reconfigurare și restabilire a stării sistemului, se poate realiza în două forme și anume:

- *reluarea*, pentru care transformarea stării eronate constă în readucerea sistemului într-una dintre stările neeronate ocupate în prealabil de către acesta înainte de apariția erorii, fie prin substituirea stării eronate printr-o stare neeronată, fie prin „derularea" acțiunilor care au avut loc de la ultima stare neeronată; respectiva stare neeronată în care este readus sistemul constituie un *punct de reluare*;
- *urmărirea*, în care caz transformarea stării eronate constă în găsirea unei *noi* stări neeronate, ce nu a survenit anterior sau nu a fost ocupată de la ultima apariție a aceleiași stări eronate;

(b) prin *compensarea erorii*, atunci cînd starea eronată conține suficientă redondanță pentru ca transformarea să poată fi efectuată pornind de la informațiile conținute în această stare eronată.

Dacă se utilizează acoperirea erorii, atunci se impune ca starea eronată să fie identificată cît mai curînd posibil ca fiind eronată — evident, înainte de a fi transformată — , acesta fiind scopul *detecției erorii*. Dimpotrivă, compensarea erorii poate fi aplicată în mod sistematic chiar și în *absența* erorii; în cazul apariției erorii ea va conduce la *mascarea* acesteia.

Rezultă că pot fi evidențiate două mari categorii de metode de tolerare a erorilor: prin *detecția* și *acoperirea erorilor* și, respectiv, prin *mascarea erorilor*.

Abordarea mascării erorilor este corespunzătoare pentru orice aplicație în timp real, dar presupune — în general — multiplicarea părții hardware a unui sistem, mai ales cînd erorile constituie cauza unor defecte fizice. Aceasta determină o creștere a costului sistemului și — privită din acest punct de vedere — metoda restabilirii sistemului la apariția unei erori (*error recovery*) apare ca fiind o abordare atractivă pentru multe aplicații și mai ales pentru sistemele distribuite. În acest caz o strategie de tolerare a erorilor se poate desfășura după cum urmează [12]: după ce a fost detectată o eroare, sarcina procesată este redistribuită între unitățile neafectate ale sistemului, iar unitatea / unitățile afectate este / sînt deconectate; în situația menționată sistemul este reîntors la o stare anterioară corectă, iar programul / programele este / sînt reluat(e).

Utilizarea singură, fără a fi însoțită de detecția erorilor, a mascării erorilor poate conduce la o diminuare sensibilă a redondanței disponibile. În concluzie, este indicat ca implementarea mascării erorilor să fie însoțită în practică de o detecție a acestora; de menționat că această detecție poate fi realizată și după mascare, tocmai datorită efectului protector al mascării.

Tratarea cauzelor erorilor implică localizarea defectărilor, pasivizarea lor și eventual reconfigurarea sistemului, dacă acesta nu mai este capabil să asigure un serviciu identic cu cel asigurat înainte de apariția erorii.

Constrîngerile modului de lucru în timp real nu permit o întrerupere a funcționării sistemului pentru o perioadă mai lungă de timp în vederea tratării erorilor, dar nici modificarea performanțelor sistemului la apariția

unei erori. Într-un astfel de sistem o problemă importantă este menținerea nealterată a stărilor anterioare ale sistemului în vederea restabilirii stării de funcționare a acestuia.

Tehnicile de *error recovery* au fost abordate de mai mulți proiectanți de sisteme tolerante la defectări. În cele ce urmează se vor analiza succint aspectele specifice acestor tehnici. În prima parte a acestui paragraf se vor studia măsurile care pot fi adoptate într-un sistem pentru transformarea unei stări eronate într-o stare neeronată, urmînd ca în continuare să fie evidențiate mecanismele procesului de restabilire a sistemului.

În abordările următoare vor fi avute în vedere *defectele de concepție* incluse direct în produsul de software. Termenul de concepție este utilizat aici în sens larg și cuprinde de fapt diferitele etape ale dezvoltării unui produs de software: specificare, concepție, codările și validările efectuate în cadrul fiecărei etape.

Dacă se consideră un produs de software (program), defectele susceptibile să-l afecteze pot fi:

- *interne* programului, adică referitoare la datele și instrucțiunile sale;

- *externe* programului, adică privind interacțiunile sale cu alte produse de software; se disting două categorii principale de defecte externe: apelarea unui program în afara specificațiilor sale și respectiv, realizarea necorespunzătoare a serviciului procurat printr-un alt program (inclusiv a procedurilor la care face apel).

Defectele de concepție interne programului pot avea manifestări multiple și imprevizibile. Rezultă că toleranța la defectele interne poate fi aplicată tuturor defectelor de concepție și realizare care au la bază o specificație comună, nefiind deci bazată pe ipoteze particulare asupra erorilor, sau — mai exact — nu se fac ipoteze privind relația dintre *defect* și *eroare*. Toleranța la defectele externe presupune dimpotrivă identificarea apriori a naturii erorii.

Toleranța la defectele interne asigură un mijloc de evitare a defectărilor, deci de împiedicare a propagării erorilor în programul unde a apărut defectul, în timp ce toleranța la defectele externe are ca scop evitarea afectării unui (sub)program de către erorile provenind de la alte (sub)programe cu care acesta interacționează.

Tehnicile de tolerare a defectelor interne nu pot asigura protecția împotriva intrărilor eronate (în afara specificațiilor programului). În plus, ele au ca scop să evite „defectarea" (sub)programului, fără ca aceasta să fie însă posibilă în toate cazurile. Este deci necesar, ca toate celelalte (sub) programe să fie protejate contra acestor defectări, pe care ele le vor resimți ca excepții; rezultă că toleranța la defectele interne și cea la defectele externe nu numai că nu se exclud reciproc, ci sînt chiar complementare.

4.4.2. MĂSURI PENTRU RESTABILIREA FUNCȚIONĂRII SISTEMELOR LA APARIȚIA ERORILOR

Detecția erorilor într-un sistem dat impune — ca o primă măsură — eliminarea lor. Există două cazuri distincte: cel al erorilor anticipate și, respectiv, al erorilor neanticipate.

Măsurile ce pot fi utilizate pentru restabilirea funcționării sistemului în cazul erorilor anticipate sînt dependente de sistemul analizat. Astfel măsura cea mai uzuală este utilizarea redondanței în informația procesată de sistem. În sistemele de calcul sînt mult folosite codurile corectoare de erori. De exemplu, cuvintele de 16 biți din memoria semiconductoare a sistemului TANDEM sînt suplimentate cu 6 biți pentru a realiza corecția unui bit eronat și detecția a două erori. De asemenea, programul memorat de sistemul ESS No 1A utilizează un cod corector de o eroare [58].

Programele de testare [2] a structurilor de date și corecție a acestora — atunci cînd se evidențiază o eroare — sînt executate periodic sau ori de cîte ori este necesar.

O abordare viabilă pentru restabilirea funcționării sistemului în cazul erorilor neanticipate constă în modificarea stării acestuia, ceea ce poate fi considerat ca un „reset" al sistemului. Strategia principală ce poate fi folosită în acest scop plasează sistemul într-o stare predefinită — de pildă, starea inițială a acestuia. Asemenea tehnici de „ștergere" pot fi utilizate atît de sistemele hardware, cît și software.

O altă formă importantă de restabilire după erori implică restaurarea stării unui sistem sau a unei părți din aceasta într-o stare presupusă fără erori. Pentru exemplificare, în figura 4.9 este ilustrată o astfel de strategie, evidențiindu-se evoluția unui sistem care a suferit un număr de restaurări ale stărilor sale. Liniile întrerupte reprezintă o activitate abandonată, iar liniile punctate — acțiunea de restaurare a stării sistemului.

Sistemul tolerant la defectări HIVE folosește restabilirea globală la o condiție inițială, utilizîndu-se în acest scop copii *back-up* ale sistemului software și fișiere ale bazei de date.

În cazul sistemului ESS No 2 [36] principalul instrument de restabilire constă în utilizarea unui program de inițializare, responsabil pentru resetarea structurilor de date. Sînt disponibile șase nivele de restabilire. La nivelul 1 registrele hardware sînt șterse; nivelurile 2—5 apelează structurile de date corespunzătoare ștergerii. Restabilirea la nivelul 6 constă în ștergerea tuturor datelor tranzitorii în memoria operativă. Pentru ștergerea datelor asociate cu apelurile deja stabilite este cerută o intervenție manuală.

De subliniat că, pentru ca o tehnică de restabilire să permită restaurarea unei stări anterioare a sistemului, este necesar să existe o înregistrare a acelei stări. Deoarece prin restaurarea unei stări anterioare se încearcă să se simuleze o „reîntoarcere" în timp, aceste tehnici de restabilire sînt

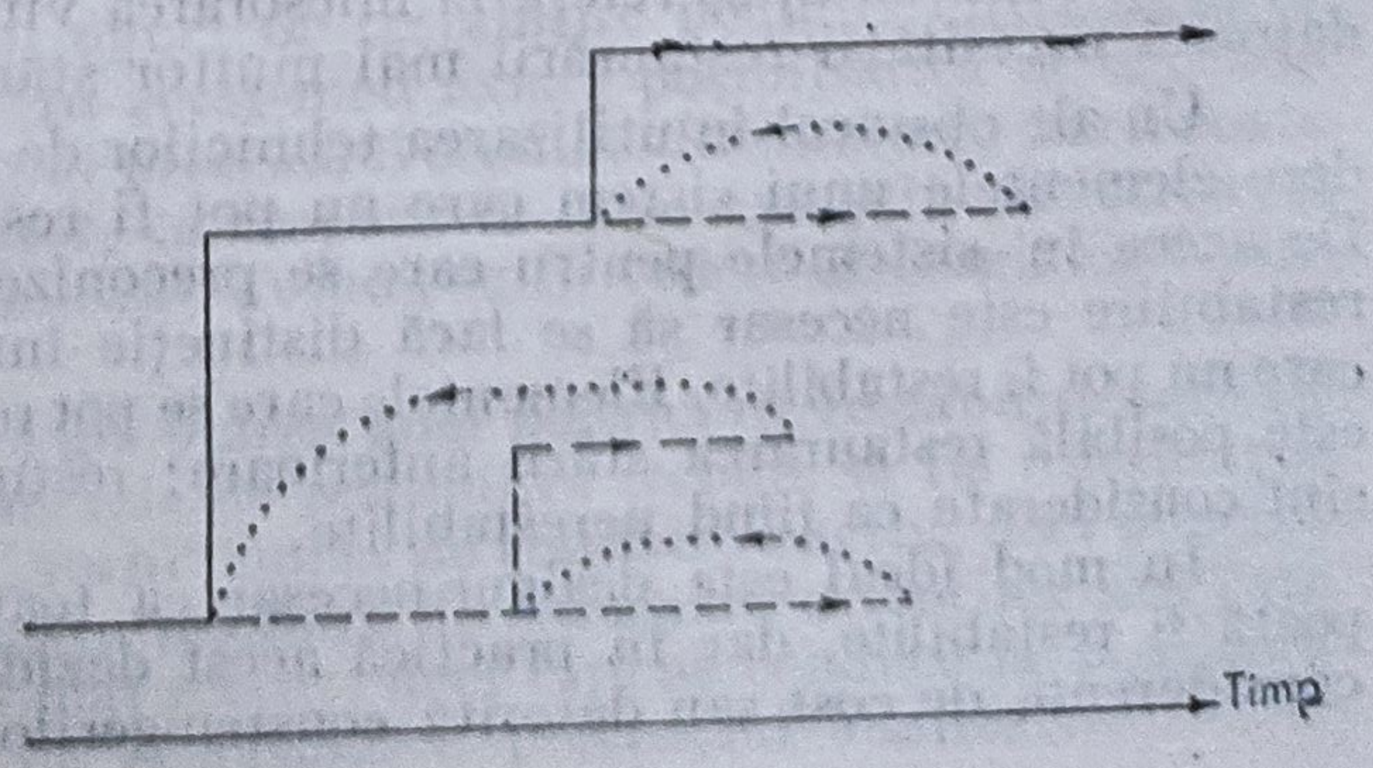


Fig. 4.9. Secvențe ale desfășurării unui proces de restabilire inversă.

denumite în literatura de specialitate *backward recovery*. Celelalte tehnici de restabilire rezultate prin îndepărtarea stării eronate se numesc tehnici de tip *forward recovery*.

Tehnicile de restabilire de tip *backward recovery* — restabilirea inversă — refac o stare anterioară a sistemului fără a mai lua în considerare starea curentă a acestuia. În contrast, tehnicile de tip *forward recovery* — restabilire directă — aduc sistemul într-o stare apropiată celei curente. Aceste tehnici prezintă următoarele caracteristici:

— sînt dependente de identificarea stării eronate;

— sînt nepotrivite pentru restabilirea sistemului la defectările neanticipate;

— sînt specifice pentru fiecare sistem în parte.

Evident, caracteristicile tehnicilor de restabilire inversă vor fi diferite:

— sînt independente de evaluarea stării eronate a sistemului;

— sînt capabile să realizeze restabilirea sistemului în prezența unor erori arbitrare;

— sînt independente de structura sistemului.

Restabilirea după apariția erorii realizează o înlocuire completă a stării sistemului, ceea ce înseamnă că evaluarea stării eronate a sistemului este necesară.

Aserțiunea conform căreia o tehnică de restabilire poate garanta revenirea sistemului dintr-o stare eronată oricare ar fi defectul este desigur exagerată. Există o clauză ce trebuie să fie îndeplinită pentru ca tehnicile de restabilire inversă să fie într-adevăr eficiente; în acest sens este necesar ca starea anterioară să poată fi restaurată cu succes, iar starea restaurată să fie anterioară manifestării defectului care generează erorile. Este clar că o astfel de restaurare va elimina toate erorile produse de defect, dacă starea anterioară restaurată va fi fără erori.

Cea mai importantă din cele trei caracteristici ale tehnicilor de tip restabilire inversă este capabilitatea de a procesa acțiuni corecte fără a ține seama de tipul defectului. Această situație permite restabilirea funcționării sistemului inclusiv în prezența erorilor de proiectare, care ar fi dificil de tolerat folosindu-se o altă metodă.

În continuare se vor prezenta mai detaliat caracteristicile tehnicilor de tip restabilire inversă.

Există unele dezavantaje ale acestor tehnici. Un prim dezavantaj constă în costul lor relativ ridicat. O tehnică de restabilire directă va costa mai puțin pentru că se restaurează numai partea eronată din starea sistemului. Al doilea dezavantaj se referă la micșorarea vitezei de operare a sistemului, datorită necesității restaurării mai multor stări anterioare ale acestuia.

Un alt obstacol în utilizarea tehnicilor de restabilire inversă are în vedere elementele unui sistem care nu pot fi restaurate la starea anterioară. De aceea în sistemele pentru care se preconizează utilizarea acestui tip de restabilire este necesar să se facă distincție între elementele care pot și cele care nu pot fi restabilite. Elementele care se pot restabili sînt acelea pentru care este posibilă restaurarea stării anterioare; restul de elemente ale sistemului sînt considerate ca fiind nerestabilite.

În mod ideal este desigur necesar ca toate elementele unui sistem să poată fi restabilite, dar în practică acest deziderat nu poate fi realizat din considerente de cost sau datorită constrîngerilor fizice.

4.4.3. TEHNICI DE RESTABILIRE A SISTEMULUI LA APARIȚIA ERORILOR

După cum s-a menționat, o abordare viabilă pentru a restabili starea sistemelor la apariția crorilor neanticipate constă în ștergerea stării curente, eronate, și reîntoarcerea sistemului la o stare anterioară lipsită de erori.

Restaurarea stării de funcționare poate fi implementată într-o varietate de căi, fiecare oferind diferite caracteristici în funcție de cost; timp de execuție, flexibilitatea sistemului, securitate etc.

În cele ce urmează se va folosi termenul de punct de restabilire pentru a denumi momentul din desfășurarea unui proces în care starea curentă a acestuia impune restaurarea sa datorită detecției unei erori; înlocuirea stării curente a unui proces cu starea pe care acesta o ocupă la un punct de restabilire va fi denumită restaurarea punctului de restabilire. Un punct de restabilire este *stabilit*, asigurînd că informația corespunzătoare este protejată, astfel încît la orice moment de timp va fi posibil să se restaureze punctul de restabilire; informația respectivă constituie *datele de restabilire*.

Un mecanism de restabilire inversă necesită disponibilitatea mai multor puncte de restabilire, iar pentru fiecare punct de restabilire va fi necesară stocarea datelor de restabilire corespunzătoare. Deoarece în acest caz costul menținerii datelor de restabilire devine ridicat, se adoptă soluția [39] ca, pe măsură ce un punct de restabilire este utilizat, să fie apoi eliminat. Odată ce un punct de restabilire a fost eliminat, nu mai este posibil ca el să mai fie restaurat. Evident, dacă toate punctele de restabilire au fost eliminate nu mai este posibilă restabilirea după o defectare a sistemului.

Un punct de restabilire este considerat *activ* din momentul stabilirii sale pînă ce este eliminat. Perioada cînd un punct de restabilire este activ se numește *regiune de restabilire*. Regiunile de restabilire se pot suprapune, așa cum este ilustrat în figura 4.10.

Pentru realizarea unui control complet și explicit al mecanismului de restabilire, este necesar să existe un *interpretor* care să asigure operațiile de stabilire, eliminare și, respectiv, de restaurare a punctelor de restabilire.

Un avantaj al utilizării restabilirii inverse constă și în faptul că proiectantul sistemului software poate prevedea punctele de restabilire la momentele procesului pe care acesta le consideră ca fiind cele mai utile.

În mod similar, punctele de restabilire pot fi eliminate atunci cînd nu mai sînt utile.

Concret, restaurarea unui punct de restabilire poate fi inițializată de către un analizor de excepții. În acest mod există posibilitatea de a corela restabilirea cu structura și scopul procesului studiat. Restaurarea stării poate fi inițiată și de către interpretor, ca răspuns la orice excepții ale interfeței considerate potențial periculoase sau avîndu-se în vedere strategia de restabilire utilizată.

În acest caz este necesară adoptarea unei politici de restabilire care să stabilească ordinea în care punctele de restabilire trebuie să fie restaurate: de exemplu, care este cel mai „vechi" punct de restabilire activ, deci punctul care va conduce la probabilitatea maximă de eliminare a erorilor sau care este punctul de restabilire cel mai recent, deci care ar conduce la pierderea minimă de activități legate de desfășurarea procesului analizat.

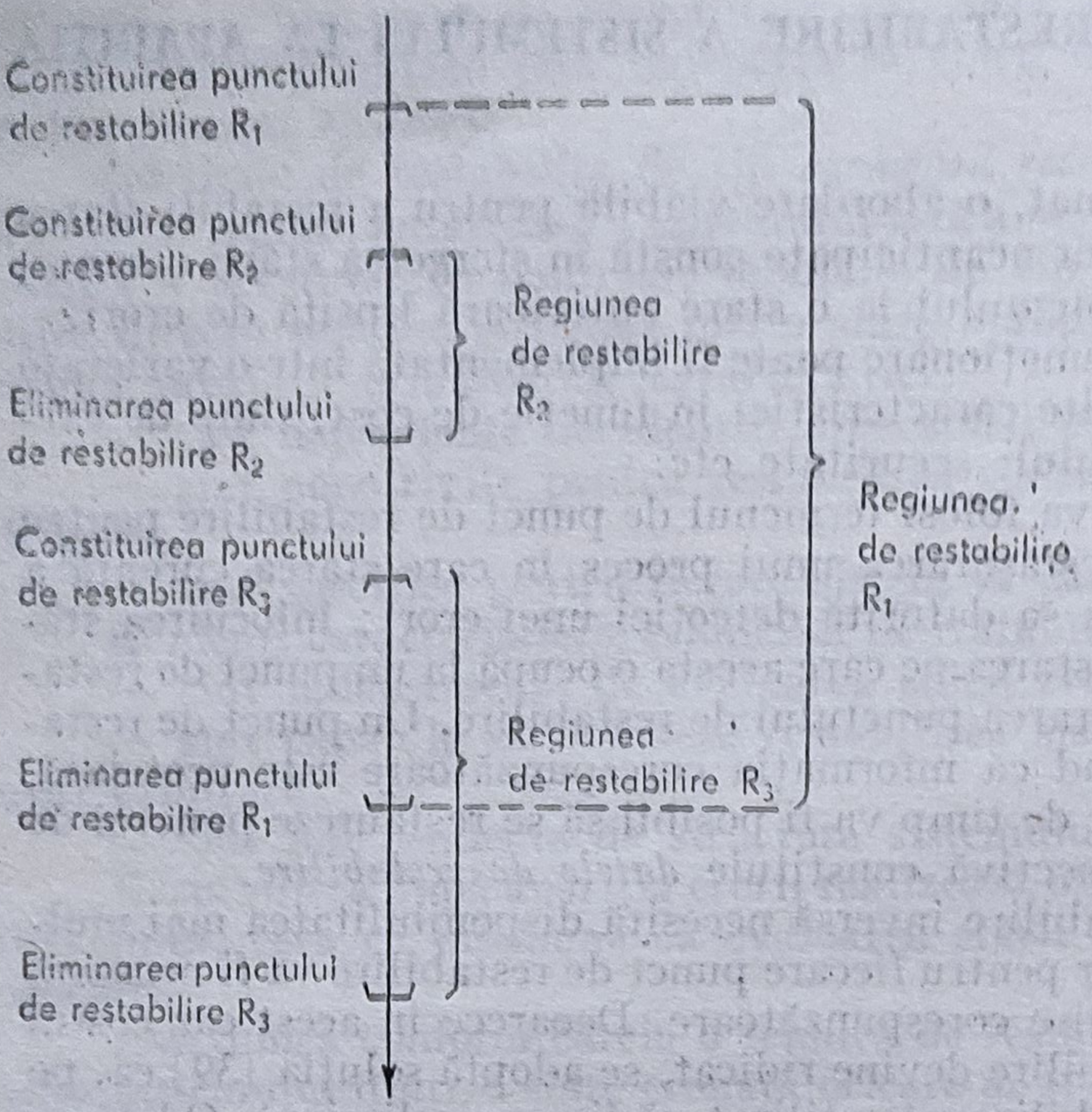


Fig. 4.10. Desfășurarea regiunilor de restabilire.

Este posibil să se construiască un mecanism de restabilire a sistemului la apariția erorilor care operează independent de utilizarea proceselor, în care caz stabilirea, restaurarea și eliminarea punctelor de restabilire se vor afla în întregime sub controlul interpretorului. De exemplu, interpretorul poate stabili în mod automat aceste puncte la intervale fixate în timp. Deși o schemă complet automată a procesului restabilirii are avantajul de a fi „transparentă" utilizatorului, ea nu permite întotdeauna o restaurare perfect concordantă cu structura proceselor analizate.

Dacă punctele de restabilire trebuie fixate la anumite intervale de timp, devine necesară stabilirea duratei acestor intervale. De pildă, fixarea punctelor de restabilire la intervale scurte de timp va conduce la cheltuirea unei cantități mari de timp pentru recepționarea datelor de restabilire. În schimb, dacă intervalul de timp între punctele de restabilire este prea mare, timpul necesar pentru restabilirea procesului și reprocesare poate fi excesiv de mare.

Din motivele expuse, devine necesară stabilirea mărimii intervalului de timp între punctele de restabilire, astfel încît timpul total necesar pentru recepționarea datelor de restabilire, restabilirea funcționării sistemului și reprocesare să fie minim. Notîndu-se cu T_r intervalul de timp între punctele de restabilire, se poate scrie [2]:

$$T_r = (2tT)^{1/2},$$

unde t este timpul de constituire a punctelor de restabilire, iar T — timpul mediu între restaurările punctelor de restabilire.

4.4.4. BLOCUL DE RESTABILIRE

Blocul de restabilire a fost introdus [33] ca o metodă de realizare a toleranței la defectări la nivel software în sistemele de calcul, în particular pentru asigurarea toleranței la erorile reziduale de proiectare. Datorită caracterului imprevizibil al manifestării erorilor de concepție interne pentru un program, toleranța lor se bazează pe existența a cel puțin două variante ale programului, obținute pornind de la aceeași specificație prin implementări *independente* — deci, utilizîndu-se concepții diversificate — și un *procedeu de decizie* destinat să furnizeze un rezultat neeronat atîta timp cît numărul defectelor nu depășește gradul de toleranță prevăzut. Blocul de restabilire constituie așadar un mijloc de implementare a redondanței la nivel software într-un sistem, cu ajutorul algoritmilor de tip *stand-by*, utilizați în vederea înlocuirii — dacă este necesar — a algoritmilor cu erori. Schema unui bloc de restabilire este prezentată în figura 4.11. Componentele de bază ale blocului de restabilire reprezintă un număr de algoritmi, denumiți *alternante*, și un *test de acceptare* care semnifică procedeul de decizie.

Blocurile de restabilire constituie astfel un procedeu de tolerare a defectărilor prin *detecție și acoperire*, acoperirea erorilor avînd loc prin *reluare*.

În schema din figura 4.11, prima alternantă reprezintă algoritmul preferat. Testul de acceptare constituie în fapt mecanismul de detecție a erorilor (utilizat de către programator) pentru testarea (verificarea) acceptabilității rezultatelor produse de către alternante. Schema blocului de restabilire se implementează ușor în limbaje structurate, ca ALGOL, PL/I, dar pot fi lesne utilizate și alte limbaje de programare, inclusiv cele de nivel înalt.

În figura 4.12 este indicată o sintaxă posibilă pentru blocurile de restabilire. Alternanta primară nu reprezintă altceva decît un bloc al programului convențional echivalent, executat pentru îndeplinirea operației căutate. *Testul de acceptare* — o expresie logică — este evaluat la ieșirea fiecărei alternante pentru a determina dacă alternanta s-a comportat de o manieră acceptabilă.

```
ensure      < test de acceptare >
by          < prima alternantă >
else by     < a doua alternantă >
. . .
. . .
else by     < a n-a alternantă >
else error
```

Fig. 4.11. Schema blocului de restabilire.

```
< bloc de restabilire >   = ensure < test de acceptare > by
                            < alternanta primară >
                            < altă alternantă >
                            else error
< alternanta primară >    = < alternanta >
< altă alternantă >       = <vidare> <altă alternantă>
                            else by <alternanta>
<alternanta>              = <lista de instrucțiuni>
<test de acceptare>       = <expresie logică>
```

Fig. 4.12. Exemplu de sintaxă pentru blocurile de restabilire.

O alternantă ulterioară — dacă există — va fi executată cînd alternanta precedentă nu a putut fi terminată (de exemplu, dacă ea comportă o împărțire prin zero, execuția sa depășește o anumită limită de timp etc.) sau dacă nu satisface testul de acceptare. Totuși, înainte ca o alternantă să nu fie executată, procesul este readus în starea pe care o ocupă înaintea execuției alternantei primare, datorită punctului de restabilire fixat la intrarea blocului de restabilire. În figura 4.13 este ilustrată execuția unui bloc de restabilire.

Execuția unui bloc de restabilire se desfășoară după cum urmează:

— se execută prima alternantă;

— la sfîrșitul alternantei este evaluat testul de acceptare; dacă acesta rezultă ca „adevărat", adică rezultatele alternantei sînt acceptabile, atunci blocul de restabilire va fi eliminat. Cînd rezultatul testului de acceptare este „fals" sau dacă este detectată o eroare în sistemul de bază, atunci în timpul execuției alternantei are loc o restabilire „inversă": starea programului este adusă în mod automat la starea existentă înaintea introducerii blocului de restabilire.

În continuare se repetă secvența descrisă anterior, cu excepția faptului că în locul alternantei eronate este introdusă cea de-a doua alternantă ș.a.m.d.

Sistemul de bază, care rulează programele conținînd blocuri de restabilire, are în structură mecanismele de control al comutării algoritmilor, per-

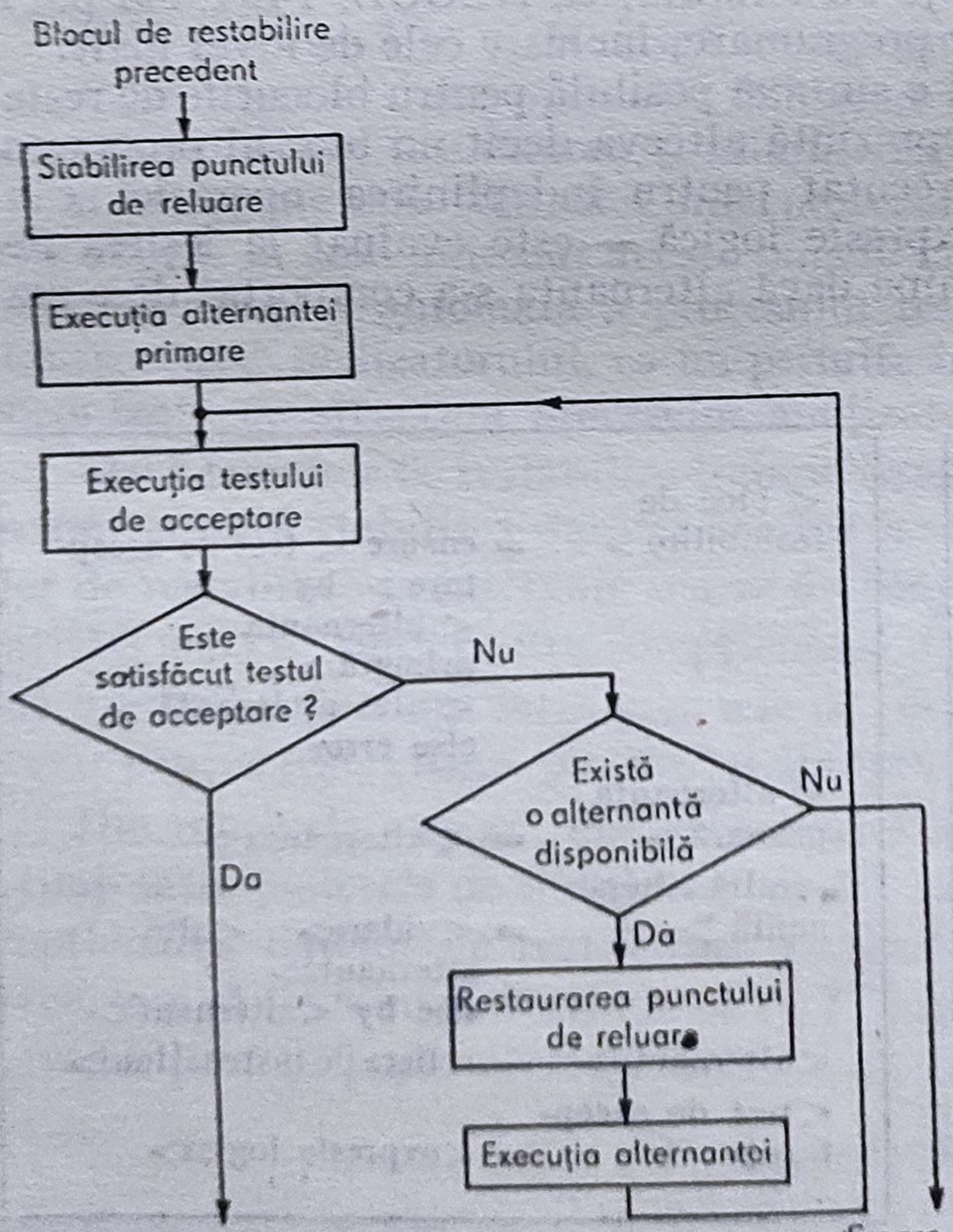


Fig. 4.13. Schema de execuție a unui bloc de restabilire.

miținд deci realizarea restabilirii „inverse". Este necesar ca aceste mecanisme să fie „transparente" programului, ele putînd fi de exemplu, implementate prin hardware-ul sistemului.

În [33] a fost propus un astfel de mecanism, denumit *recovery cache*, care permite realizarea unei restabiliri „inverse". Acesta realizează în principal trei funcții: înregistrarea datelor privind restabilirea, restabilirea propriu-zisă și eliminarea punctelor de restabilire atunci cînd acestea nu mai sînt necesare.

În continuare vor fi evidențiate unele particularități ale mecanismului de tolerare a erorilor bazat pe blocurile de restabilire.

Principial, blocurile de restabilire pot fi proiectate pentru a realiza toleranța sistemului la *defectările algoritmice*, atît la nivel hardware cît și la nivel software.

Într-un sistem, defectele se pot clasifica — conform [39] — în defecte ale componentelor fizice, cînd respectivele componente nu își îndeplinesc funcțiile prevăzute în acord cu specificațiile, și, respectiv, în *defecte algoritmice*, care privesc interconexiunile dintre componente. La nivel hardware, într-un sistem pot apărea atît defecte ale componentelor cît și defecte algoritmice, ultimele fiind — de exemplu — conexiunile lipsă sau incorecte între componente. La nivel software, defectele sînt algoritmice, ele fiind în fapt defecte de proiectare. Localizarea și efectele unor astfel de defecte nu pot fi în general anticipate din moment ce ele își au originea în complexitatea — greu de stăpînit — a proiectării unui sistem. Într-un sistem cu blocuri de restabilire aceste defecte pot fi evitate prin comutarea la următoarea alternantă, cu condiția ca mulțimea circumstanțelor care au condus la defectul respectiv în alternanta anterioară să nu mai apară. Tratarea defectului este lăsată pentru diagnoză manuală de tip *off-line*, favorizată de înregistrările de date privind erorile menționate anterior.

Precizăm și faptul că un bloc de restabilire poate de asemenea realiza toleranța sistemului și pentru unele defecte anticipate ale componentelor, atît tranzitorii cît și permanente.

În principal există două căi diferite de utilizare a unui bloc de restabilire. Prima și cea mai uzuală situație intervine atunci cînd este necesar ca fiecare alternantă a unui bloc de restabilire să conducă la exact aceleași rezultate. În acest caz alternantele trebuie realizate utilizîndu-se versiuni diferite ale aceluiași algoritm sau este necesar să fie proiectate de specialiști diferiți, în scopul evitării aceluiași tip de erori.

A doua situație corespunde cazului în care blocul de restabilire realizează o reconfigurare a software-ului de tip *graceful-degradation* [23]. În această situație nu mai este necesar ca fiecare alternantă să producă aceleași rezultate. Cerința impusă alternantelor în acest caz este ca ele să producă rezultate „admise" de testele de acceptare. Astfel, doar prima alternanta urmează să conducă la rezultatele dorite, în timp ce a doua și următoarele alternante ale blocului vor realiza doar un serviciu cu un grad de degradare crescut. Cel mai „degradat" serviciu este realizat de alternanta cea mai simplă, care va avea evident probabilitatea cea mai mică a erorilor de proiectare. Un exemplu de bloc de restabilire proiectat în această manieră este prezentat în figura 4.14. Testul de acceptare pentru acest bloc verifică numai că șirul de transfer se află într-o stare consistentă.

```
ensure  <consistența șirului de transfer pentru disc>
by      <algoritm care intră solicitat în poziția optimală a șirului>
else by <algoritm care intră solicitat la sfîrșitul șirului>
else by <trimite avertismentul solicitare ignorată>
else error
```

Fig. 4.14. Exemplu de bloc de restabilire.

Alternanta primară încearcă să plaseze noua cerință de transfer în poziție optimală în șir (de exemplu, pentru .a minimiza mișcarea capului de disc). A doua alternantă evită complicațiile primei alternante pur și simplu prin dispunerea noii cerințe la sfîrșitul șirului.

De menționat că blocul de restabilire realizează o redondanță la nivel algoritmic, dar nu și în structurile de date ale programului.

Este posibil să se facă o analogie între înlocuirea alternantelor în schema blocului de restabilire și înlocuirea componentelor defecte într-o structură redondantă de comutație implementară în hardware dar, în același timp însă trebuie avute în vedere următoarele aspecte: mai întîi, o componentă hardware este în mod uzual înlocuită cu altă componentă identică din punctul de vedere al proiectării și construcției, ceea ce nu este de obicei cazul alternantelor blocului de restabilire. În al doilea rînd, înlocuirea unei componente hardware este în general permanentă; componenta înlocuită poate fi reparată și menținută în rezervă *stand-by* atît cît este nevoie. În blocurile de restabilire alternantele defecte sînt înlocuite numai temporar și anume atîta timp cît este necesară execuția blocului; pentru un alt set de date de intrare, prima alternantă, va putea fi din nou utilizată, deoarece este posibil — și se speră — ca noul set de intrări să nu mai conducă la manifestarea defectului.

Testul de acceptare, după cum sugerează și denumirea, va fi un test de acceptabilitate rezultatelor alternantelor și mai puțin unul de verificare a corecției lor absolute. El este destinat pentru luarea unei decizii referitoare la rezultatele furnizate de către alternante.

Testele de acceptare — strîns legate de aplicația considerată — pot fi privite ca aserțiuni executabile [52]. De exemplu, cazul aplicațiilor în timp real — pentru care neobținerea rezultatelor înaintea unei anumite date poate fi considerată drept defectare — poate fi luat în considerare pentru testele de acceptare prin includerea mecanismelor de tip *Watch-dog* [2].

Alegerea unui anumit test de acceptare trebuie să fie rezultatul unui compromis între complexitatea și eficacitatea procesului de detecție pe care aceasta se bazează. Astfel, o prea mare complexitate poate conduce la o creștere substanțială a timpului de execuție și adesea face testul mai vulnerabil la defectele de concepție și codare.

Atunci cînd alternantele unui bloc de restabilire au fost proiectate pentru a produce un software reconfigurabil de tip *gracefully degradation*, este clar că testul de acceptare poate fi folosit doar pentru verificarea rezultatelor ultimei alternante, și anume cea mai „slabă" (degradată). Acest fapt a sugerat posibilitatea utilizării unor teste de acceptare separate, pentru fiecare alternan tă în parte, organizate după cum este ilustrat în figura 4.15.

```
ensure <adevărat>
by ensure <testul de acceptare cel mai bun>
    by <algoritmul cel mai bun> else error;
else by ensure <testul de acceptare următor cel mai bun>
    by <algoritmul următor cel mai bun> else error;
. . .
. . . .
else error
```

Fig. 4.15. Teste de acceptare multiple.

În general, se procedează la un singur test de acceptare, mai ales atunci cînd alternantele produc aceleași rezultate.

Cum cercetările în acest domeniu sînt încă în faza de pionierat nu există o metodologie unitară de proiectare a testelor de acceptare pentru fiecare problemă specifică, dezvoltîndu-se teste de acceptare particulare. De subliniat că testul de acceptare nu trebuie privit numai ca mecanism de detecție a erorii în sistem, ci el permite și desfășurarea algoritmilor din blocul de restabilire. Mai mult, noile teste scrise de programatori pot fi incorporate în alternante prin intermediul declarațiilor de tip *assert* [52].

Ca orice sistem care realizează o tolerare a defectărilor prin utilizarea unor structuri redondante, folosirea blocurilor de restabilire conduce la timpi de rulare superiori comparativ cu cazul programelor netolerante la defectări. Dar costurile ridicate, necesitatea de testare și validarea apriori a programelor fiabile netolerante la defectări a condus la concluzia [31] că folosirea blocurilor de restabilire constituie o metodă eficace de reducere a acestor costuri. Timpul suplimentar cerut de blocurile de restabilire depinde de timpul necesar pentru evaluarea testului de acceptare și de modul de implementare a algoritmilor blocului. În [37] se propune o arhitectură a blocului de restabilire care încearcă să micșoreze aceste supracreșteri de timp.

În încheiere, metoda blocurilor de restabilire se compară cu alte tehnici de tolerare a erorilor ce pot fi utilizate în domeniul sistemelor software.

Tratarea excepțiilor, de exemplu în structura propusă în [24], este adesea invocată ca o alternativă a blocurilor de restabilire. Dar o asemenea tratare poate asigura o restabilire eficientă doar pentru defectele specifice care pot fi analizate și ale căror consecințe pot fi prevăzute integral.

În contrast cu această metodă, restabilirea de tip „invers" implică restaurarea completă a stării sistemului, nu doar restaurarea părților eronate ale acestuia.

Totuși, metoda blocurilor de restabilire poate realiza o toleranță la defectările neanticipate, din moment ce în cazul restabilirii de tip „invers" nu este necesar să se facă nici o ipoteză despre defect și consecințele sale; rezultă că avem de-a face cu o metodă *generală* de restabilire. De subliniat că metodele blocurilor de restabilire și de tratare a excepțiilor trebuie

să fie privite mai degrabă ca fiind abordări complementare și nu concurente. De altfel, există încercări [46] de combinare a celor două tehnici menționate anterior, în scopul realizării unui software tolerant la erori eficient.

4.4.5. PROGRAMAREA *N*-VERSIONALĂ

Principiul programării *N*-versionale constă în execuția în paralel a unui număr impar de variante, denumite versiuni, și furnizarea rezultatului lor unui sistem de decizie (voter) care efectuează o *votare*, permițînd ca — atîta timp cît există numai o minoritate de versiuni defecte — să se considere rezultatul corect. Rezultă că programarea *N*-versională este o metodă de tolerare a defectărilor prin mascarea erorilor; prin analogie cu cazul sistemelor hardware, ea poate fi considerată ca o redondanță de tip static.

Comportarea unui produs de software triversional este evidențiată în figura 4.16. De menționat că reîntoarcerea spre versiuni, indicată punctat în figură, simbolizează eventualele acțiuni ce pot fi executate în caz de detecție a erorii: oprire sau reinițializarea versiunii eronate.

Controlul execuției versiunilor este asigurat prin intermediul unui (sub) program special, denumit controler. Acesta comportă structura decizională care asigură coordonarea execuției versiunilor prin intermediul mecanismelor de sincronizare și care efectuează votarea. El conține, de asemenea, mecanismele de „acoperire" a versiunilor defecte.

Pentru implementarea mecanismului decizional prezintă o importanță deosebită *vectorii de comparație*, utilizați în scopul efectuării unei votări în punctele de decizie, denumite puncte de votare, precum și indicatorii de stare destinați comparației.

Vectorii de comparație sînt structuri de date reprezentînd o submulțime a stării unei versiuni asupra căreia trebuie efectuată votarea. În plus, față de variabilele de comparație, vectorii de comparație trebuie să posede indicatori de stare, care să poată evidenția apariția anumitor evenimente, cum ar fi de exemplu excepțiile.

Indicatorii de stare pentru comparație au rolul de a indica acțiunile ce trebuie întreprinse după votare; aceste acțiuni sînt condiționate de apariția vectorilor de comparație într-un interval de timp specificat, precum și de acordul sau dezacordul vectorilor de comparație.

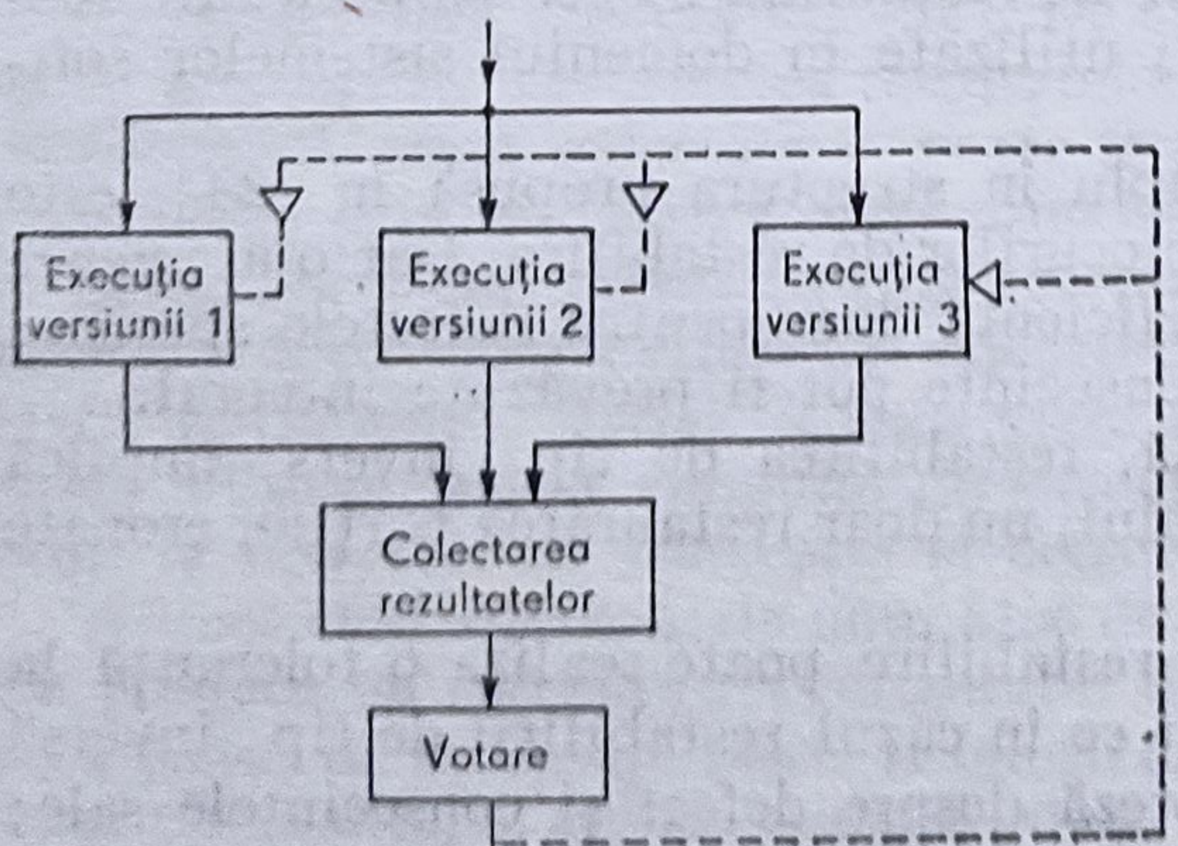


Fig. 4.16. Comportarea unui sistem software 3-versional.

Algoritmii de votare sînt specifici fiecărei aplicații. Este totuși posibil să se distingă trei mari categorii de algoritmi, în funcție de natura informațiilor asupra cărora se efectuează votarea — informații binare de tip „totul sau nimic", informații numerice și, respectiv, șiruri de caractere.

În cazul informațiilor binare, se efectuează o votare majoritară de tip clasic. În ceea ce privește informațiile numerice, nefiind posibilă în general o votare exactă, este indicat să se utilizeze procedee de votare de tipul celor folosite în cazul semnalelor analogice. Referitor la șirurile de caractere, cum sînt informațiile nenumerice date operatorului, este important să fie detectate așa-numitele erori de formă [6] (cum sînt greșelile de ortografie sau formatele eronate) înainte ca aceste erori — ușor corectabile de către operator — să aibă drept consecință declararea unei versiuni corespunzătoare ca fiind eronată.

De subliniat, că elaborarea algoritmului de votare constituie o problemă importantă în conceperea unui produs de software cu structură *N*-versională. Alegerea unui algoritm de votare trebuie să aibă în vedere atît eficacitatea, cît și simplitatea sa.

Serviciile asigurate de către diferitele versiuni trebuie să fie identice. În literatura de specialitate se consideră că este indicat ca versiunile unui produs software cu structură *N*-versională să fie realizate — pornindu-se de la o specificație comună — de către echipe de concepție diferite. Totuși, considerăm că această problemă mai necesită cercetări: în fapt, dacă pe de o parte existența mai multor echipe de programatori conduce la o mai mare diversitate a produselor realizate, pe de altă parte riscă să impună ca necesară o specificare mai detaliată a punctelor de decizie, ceea ce duce în realitate la o scădere a diversității.

Un alt aspect important ce trebuie avut în vedere se referă la coerența datelor de intrare ale diferitelor versiuni: deoarece acestea se execută în paralel, devine necesar ca datele care le sînt furnizate să fie coerente. O soluție posibilă în această direcție constă în adoptarea unor algoritmi de coerență interactivi [62], [34] de tipul celor utilizați în sistemele distribuite.

4.4.6. PROGRAMAREA *N*-AUTOTESTABILĂ

Cercetări recente privind realizarea unor sisteme software cu structură tolerantă la erori au condus la apariția unei alte abordări în acest domeniu, denumită *programarea N-autotestabilă* [38]. În acest caz, toleranța la erori este realizată prin execuția în paralel a cel puțin două componente „autotestabile". La fiecare execuție a sistemului astfel constituit o singură componentă este considerată ca fiind activă, avînd în vedere serviciul asigurat.

Celelalte componente autotestabile pot fi apreciate drept componentele unei structuri redondante active. În cazul „defectării" componentei active, livrarea (asigurarea) serviciului este realizată de către una dintre aceste componente. Tratarea erorii implică deci detecția erorii și comutarea serviciului pe o altă componentă.

Această tehnică poate fi privită — prin analogie cu tehnicile de tolerare a defectărilor hardware — ca o redondanță dinamică activă.

Se pot distinge două forme de componente autotestabile: cele constituite dintr-o variantă și un test de acceptare, respectiv cele constituite din asocierea a două variante și a unui algoritm de comparare a rezultatelor lor.

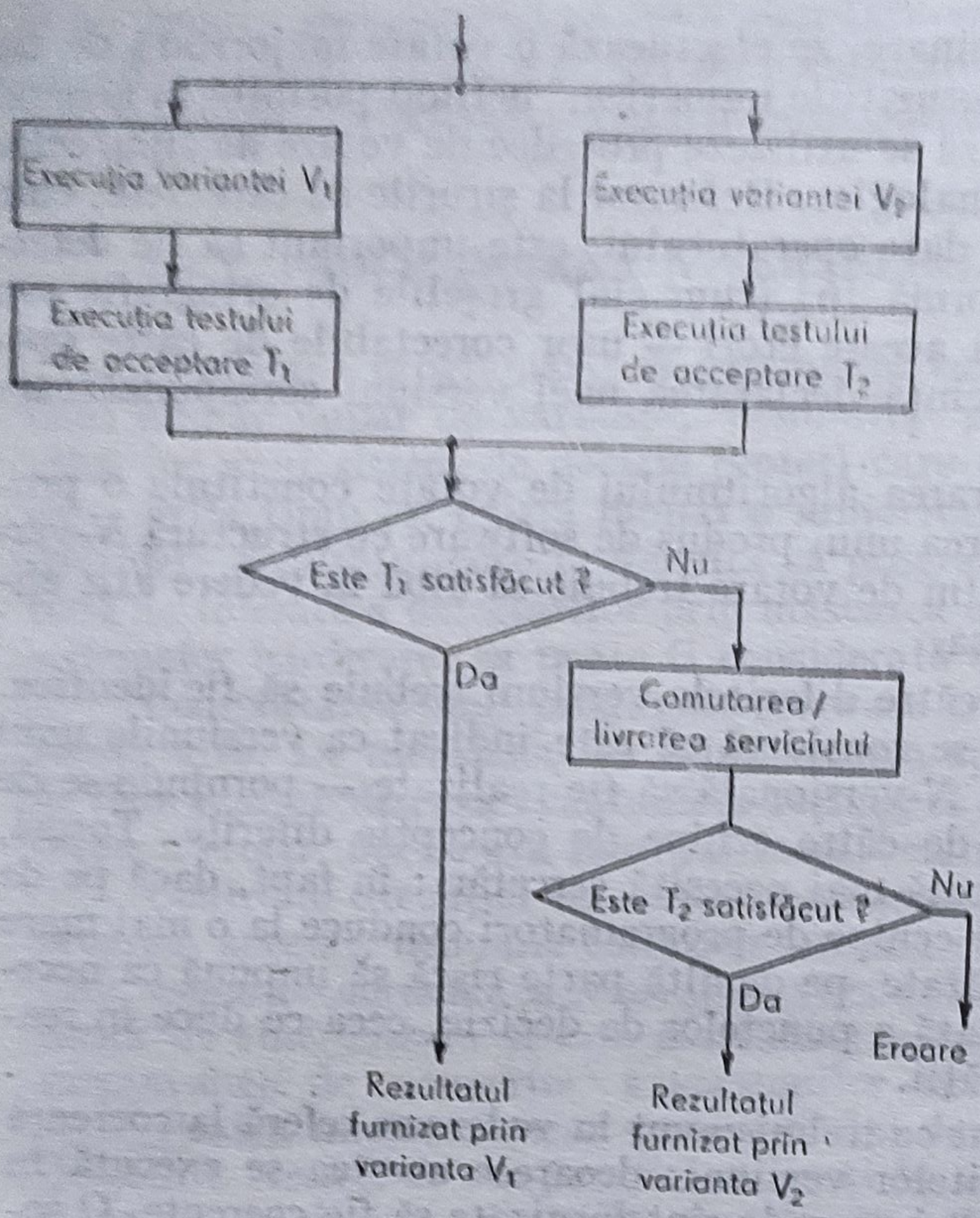


Fig. 4.17. Principiul de funcționare a unui sistem software autotestabil.

Prima formă poate fi privită ca un *bloc de restabilire*, unde alternantele sînt executate în paralel. Comportarea unui astfel de sistem software tolerant la erori în structura 2-versională este indicată în figura 4.17.

În a doua formă, dacă compararea rezultatelor furnizate de către cele două variante componente este satisfăcătoare, rezultatele puse la dispoziție de componenta autotestabilă provin dintr-una din cele două variante.

Această a doua formă poate fi considerată ca o formă specială a primei: varianta care nu furnizează rezultate în exteriorul componentei autotestabile și algoritmul de comparație pot fi privite drept teste de acceptare.

Comportarea unui sistem software cu structură 2-versională, pentru care componentele autotestabile sînt formate din două variante și un algoritm de comparare a rezultatelor lor este indicată în figura 4.18.

Ca și în cazul programării *N*-versionale, datorită faptului că aceste componente software sînt executate în paralel, este necesar un mecanism de coerență a intrărilor [34].

4.4.7. ANALIZĂ COMPARATIVĂ A TEHNICILOR DE REALIZARE A SOFTWARE-ULUI CU STRUCTURĂ TOLERANTĂ LA ERORI

Oricare ar fi tehnica utilizată, concepția unui sistem software cu structură tolerantă la erori se bazează pe o metodologie comună (fig. 4.19).

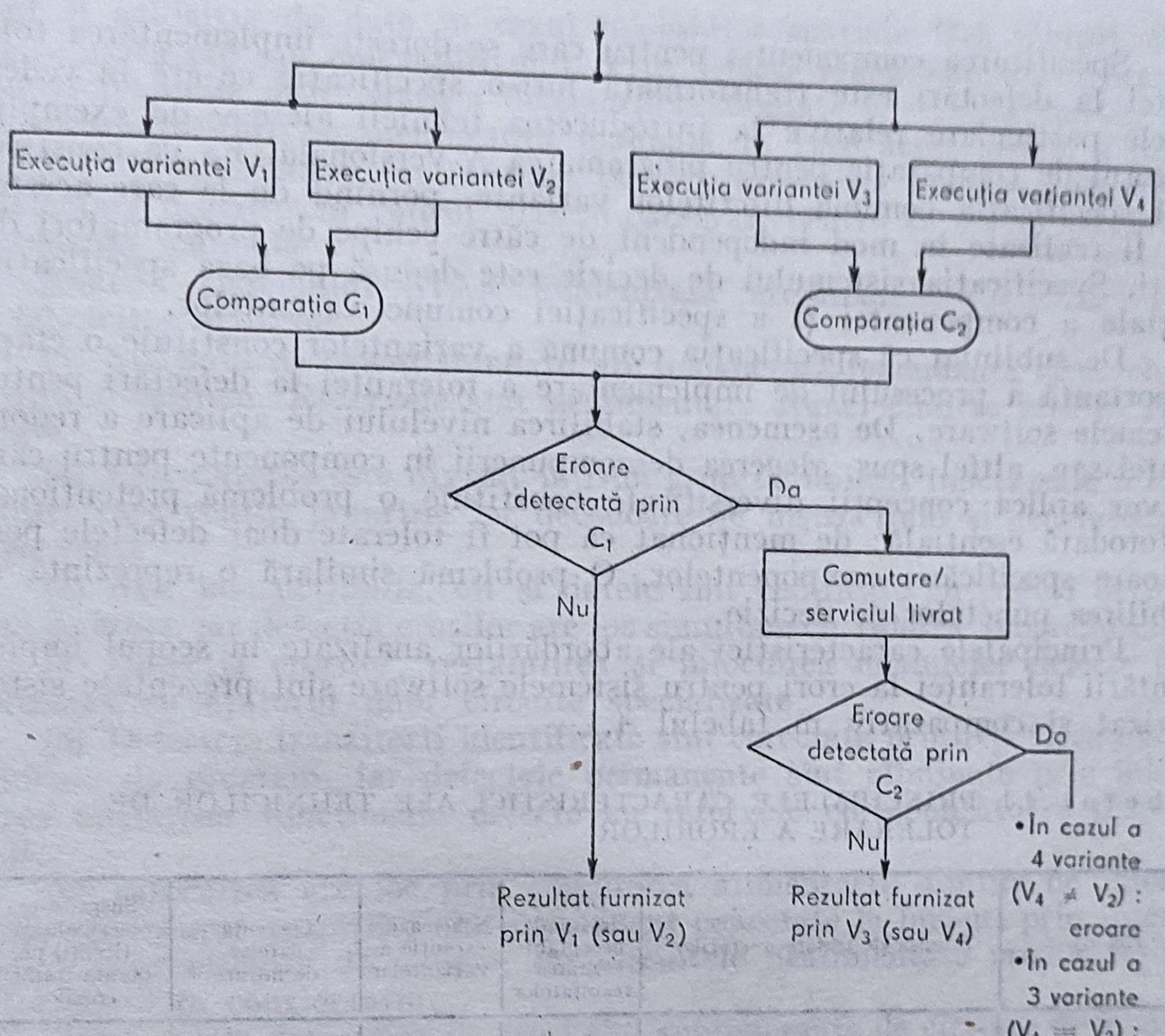


Fig. 4.18. Principiul de funcționare a unui sistem software autotestabil cu algoritm de comparare.

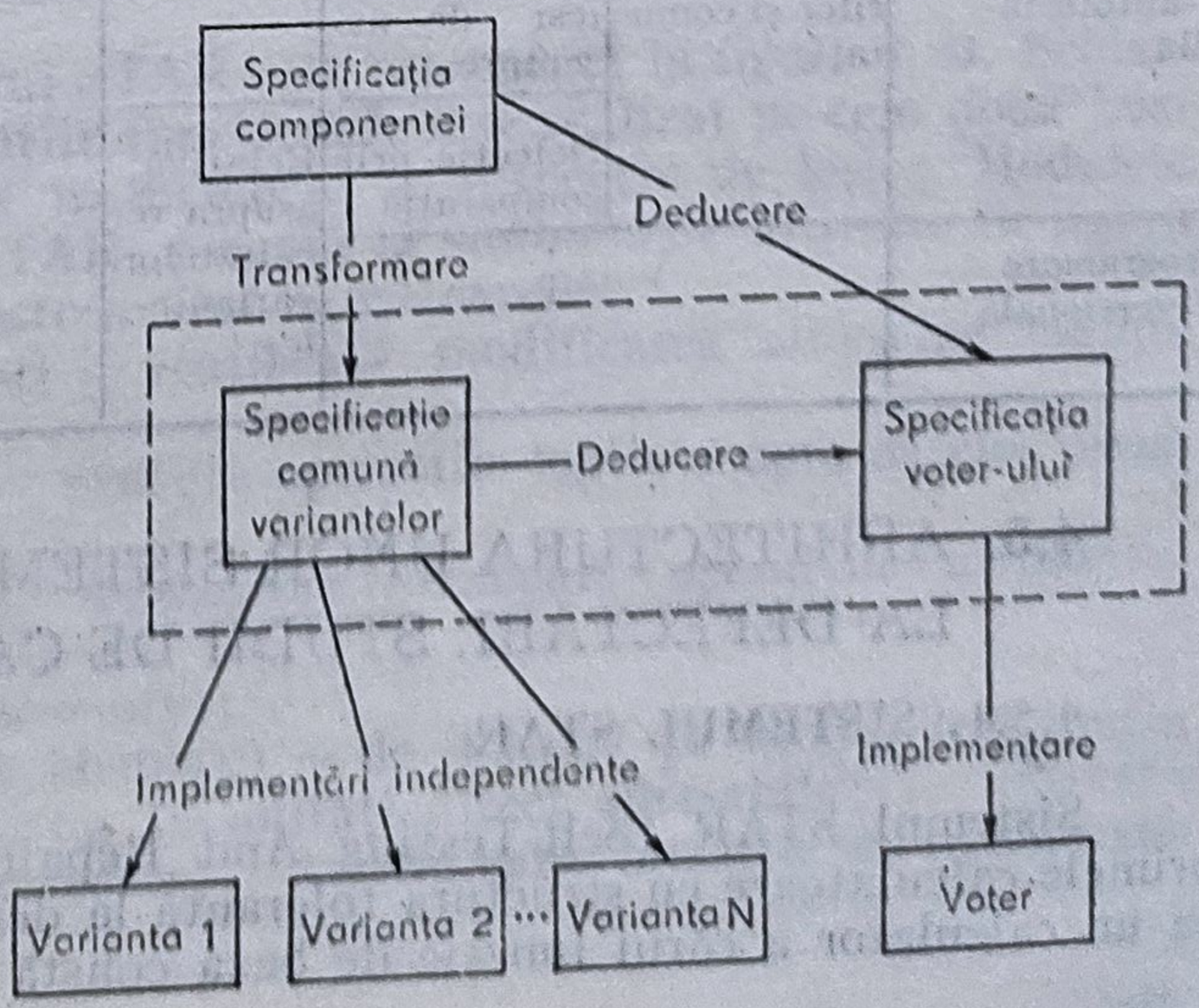


Fig. 4.19. Procesul de realizare a unui sistem software tolerant la erori.

Specificarea componentei pentru care se dorește implementarea toleranței la defectări este transformată într-o specificație ce are în vedere datele particulare relative la introducerea tehnicii alese — de exemplu, vectorul de comparație pentru programarea *N*-versională. Ea va constitui deci specificația comună diferitelor variante, pornind de la care acestea vor fi realizate în mod independent de către echipe de programatori diferiți. Specificația sistemului de decizie este dedusă pe baza specificației inițiale a componentei și a specificației comune variantelor.

De subliniat că specificația comună a variantelor constituie o etapă importantă a procesului de implementare a toleranței la defectări pentru sistemele software. De asemenea, stabilirea nivelului de aplicare a redondanței sau, altfel spus, alegerea descompunerii în componente pentru care se vor aplica concepții diversificate constituie o problemă pretențioasă și totodată esențială: de menționat că pot fi tolerate doar defectele posterioare specificării componentelor. O problemă similară o reprezintă și stabilirea punctelor de decizie.

Principalele caracteristici ale abordărilor analizate în scopul implementării toleranței la erori pentru sistemele software sînt prezentate sistematizat și comparativ în tabelul 4.1.

Tabelul 4.1. PRINCIPALELE CARACTERISTICI ALE TEHNICILOR DE TOLERARE A ERORILOR

<table>
<tr><th>Metoda</th><th colspan="2">Tehnica de tratare a erorilor</th><th>Decizia privind acceptabilitatea rezultatelor</th><th>Schema de execuție a variantelor</th><th>Coerența datelor de intrare</th><th>Suspendarea serviciului (livrat) pe durata tratării erorilor</th></tr>
<tr><td>Blocuri de restabilire</td><td colspan="2">Detecția erorilor prin test de acceptare și rulare</td><td>Absolută în raport cu specificația</td><td>Secvențială</td><td>Implicită, prin punctele de reluare</td><td>Da, pe durata execuției uneia sau mai multor variante</td></tr>
<tr><td rowspan="2">Programare N-autotestabilă</td><td rowspan="2">Detecția erorilor și comutație</td><td>Detecție prin test de acceptare</td><td rowspan="2"></td><td rowspan="2">Paralelă</td><td rowspan="2">Explicită, prin mecanisme corespunzătoare</td><td>Da, pe durata comutației</td></tr>
<tr><td>Detecție prin comparație</td><td>Relativă, asupra rezultatelor variantelor</td><td>Nu</td></tr>
<tr><td>Programare N-versională</td><td colspan="2">Votare</td></tr>
</table>

4.5. ARHITECTURA UNOR SISTEME TOLERANTE LA DEFECTĂRI. STUDII DE CAZ

4.5.1. SISTEMUL STAR

Sistemul STAR (Self-Testing And Repairing) [4] este unul dintre primele calculatoare cu structura tolerantă la defectări. El a fost conceput ca un calculator a cărui funcție de bază constă în ghidarea spațială, con-

trolul și achiziția de date în cazul misiunilor spațiale fără echipaj, de lungă durată (10 sau mai mulți ani). Prin urmare redondanța încorporată — ca și tehnicile de tolerare a defectărilor adoptate — reflectă condițiile fără disponibilități de mentenanță manuală în care acest sistem a fost prevăzut să funcționeze, în contrast cu structura altor sisteme tolerante la defectări descrise în cadrul acestui paragraf, pentru care este posibilă mentenanța manuală.

Analiza sistemului STAR evidențiază următoarele particularități ale tehnicilor utilizate:

(1) Sistemul de calcul standard are o structură redondantă la nivelul fiecărui subsistem. Rezervele sînt nealimentate atunci cînd se află în așteptare;

(2) Calculatorul este divizat într-un grup de unități funcționale înlocuibile, care conțin propriile lor decodoare de instrucțiuni și generatoare de secvențe;

(3) Atît instrucțiunile, cît și datele sînt codificate cu coduri detectoare de erori, iar detecția erorilor are loc simultan cu rularea programului;

(4) Detecția erorilor, restabilirea și înlocuirea modulelor defecte se realizează cu ajutorul unor circuite specializate;

(5) Defectele tranzitorii identificate sînt corectate prin repetarea unor segmente de program, iar defectele permanente sînt eliminate prin înlocuirea unităților funcționale defecte cu rezervele corespunzătoare disponibile;

(6) Înlocuirea are loc prin comutarea alimentării. Liniile de informație ale tuturor unităților sînt permanent conectate la bus-uri prin intermediul unor circuite de izolare, iar circuitele nealimentate produc ieșiri „0“ neluate în considerație;

(7) Codurile detectoare de erori sînt suplimentate de circuite de monitorizoare, care controlează funcționarea internă a unităților funcționale și sincronizarea alimentării;

(8) Circuitele specializate pentru detecția defectărilor și restabilirea funcționării modulelor defecte — care alcătuiesc procesorul TARP — sînt triplate, și în plus, există posibilitatea de înlocuire a unității defecte din triplet.

Schema-bloc a sistemului STAR este prezentată în figura 4.20. Schimbul de informație între unitățile funcționale este realizat pe cele două bus-uri, *M-I* și *M-O*, de intrare în memorie și, respectiv de ieșire. Modulele funcționale ale sistemului STAR, precum și simbolurile utilizate în figura 4.20 pentru desemnarea acestora sînt următoarele:

COP (Control Procesor) — realizează modificarea adreselor înainte de execuție;

LOP (Logic Procesor) — execută operațiile logice asupra datelor (două copii sînt alimentate);

MAP (Main Arithmetic Processor) — execută operațiile aritmetice asupra datelor;

ROM (READ-ONLY Memory);

RWM (READ-WRITE Memory) — de 4 Kcuv pe modul (cel puțin două copii sînt alimentate), cu 12 unități direct adresabile;

IOP (Input/Output Processor) — controlează intrările și ieșirile sistemului;

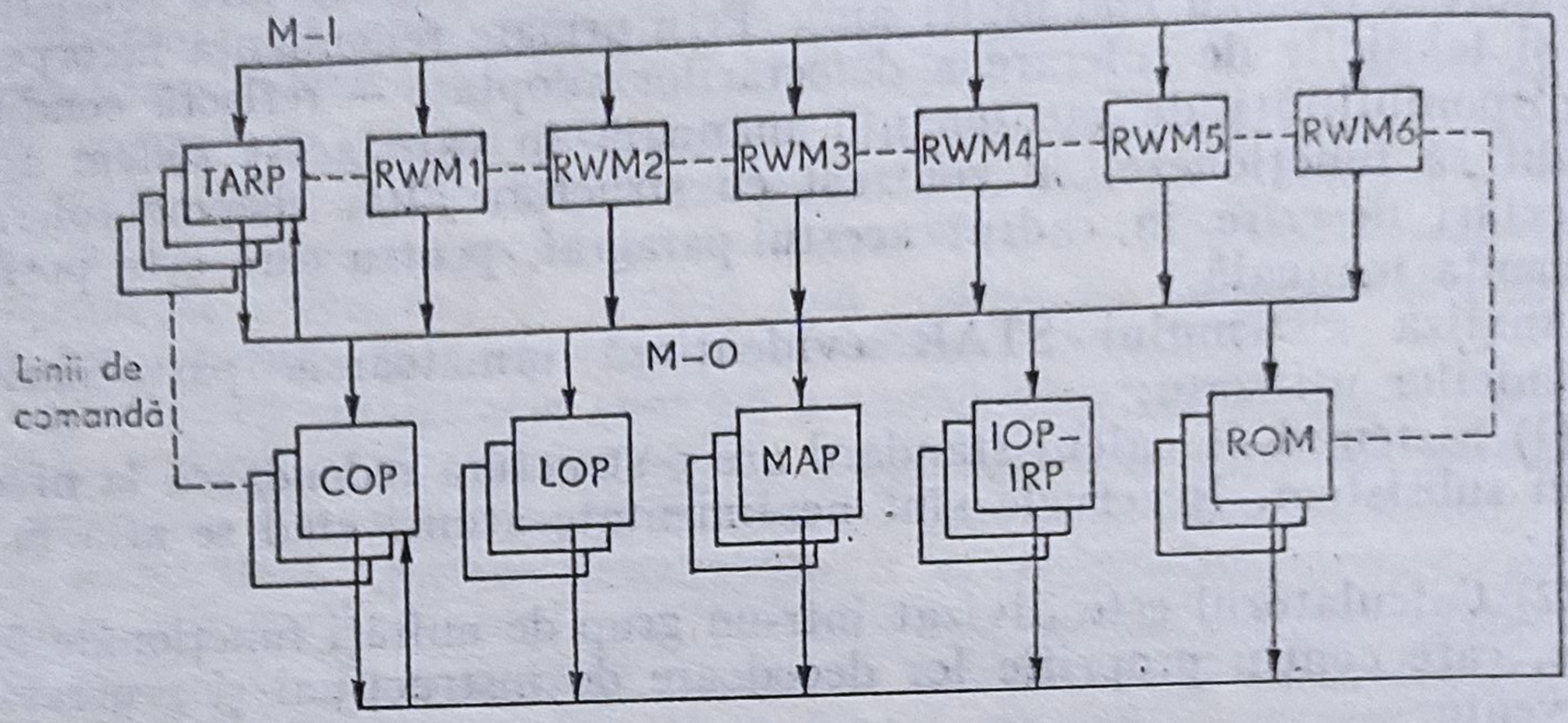


Fig. 4.20. Schema organizării calculatorului tolerant la defectări STAR.

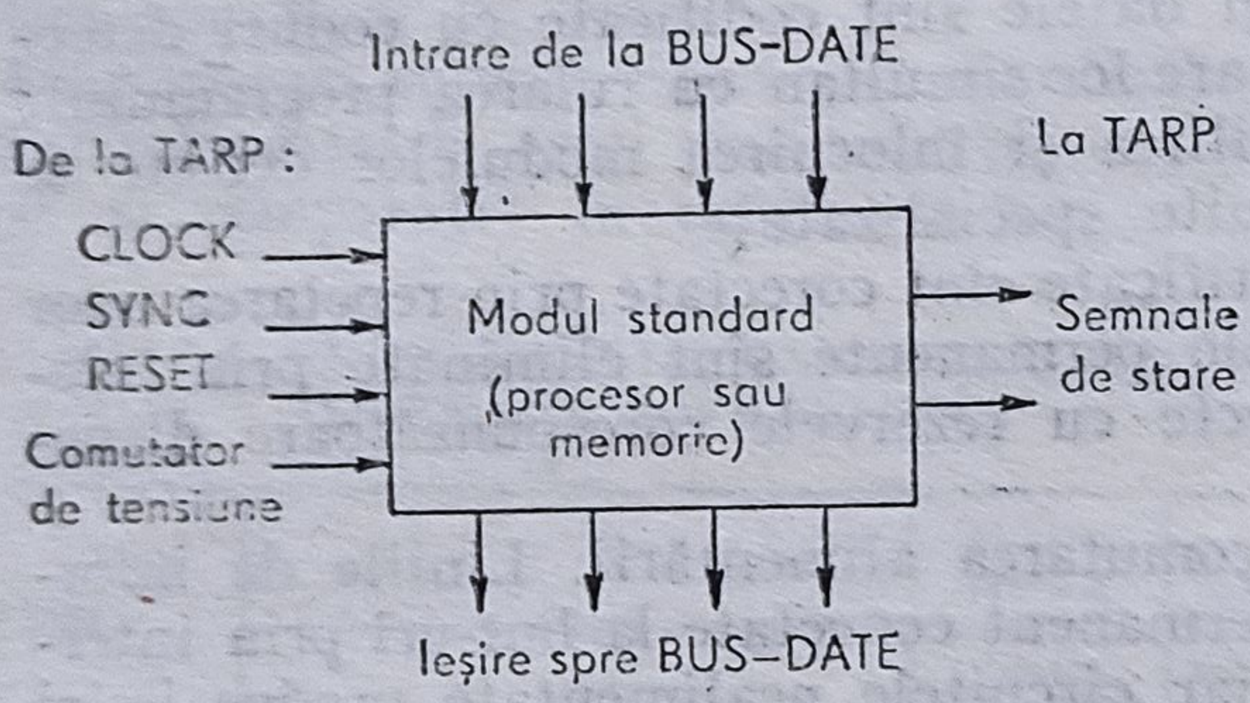


Fig. 4.21. Interfața unei unități funcționale din calculatorul STAR.

IRP (Interrupt Processor) — tratează cererile „interrupt";

TARP (Test And Repair Processor) — monitorizează funcționarea calculatorului și implementează restabilirea sistemului. Are o structură redondantă de tip hibrid.

Procesorul TARP trimite un grup de trei semnale de control pe bus-ul datelor de control pentru sincronizarea funcționării unităților și inițializarea restabilirii sistemului.

Un modul (memorie sau procesor) are o interfață într-o configurație standard (fig. 4.21) formată din 14 linii pentru semnale: 11 linii sînt permanent conectate la sistemul de bus-uri al calculatorului, iar trei linii sînt conectate permanent la TARP. În figura 4.21 este indicată distribuția acestor linii:

— patru linii de intrare și patru de ieșire ale interfeței sînt conectate la M-*I* și *M-O*. Ele recepționează și transmit cuvintele în cod mașină;

— o linie este pentru comutarea tensiunii de alimentare atunci cînd unitatea funcțională trece din rezervă în starea activă;

— trei linii sînt pentru cele trei semnale de intrare pe magistrala de control: CLOCK, SYNC — un semnal periodic de sincronizare și RESET — un semnal care forțează unitatea într-o stare inițială standard;

— două linii spre procesorul TARP trimit informația privind starea unității respective.

Fiecare unitate funcțională este autonomă și conține propriul său generator de secvențe, ca și posibilitatea de memorare a codului operațiilor curente, operanzilor și rezultatelor.

Funcționarea corectă a sistemului este determinată de procesorul TARP, care furnizează și primește toate semnalele de analiză a modului de funcționare a unităților și bus-urilor. Sub controlul său sistemul prezintă două moduri de operare: *standard* și *recovery*.

În timpul operării *standard* procesorul TARP emite semnalele CLOCK și SYNC, atunci cînd este inițiat un nou pas în execuția unei instrucțiuni. Zece perioade ale semnalului CLOCK formează baza de timp a calculatorului. În timpul primei perioade un cuvînt de patru biți, *step code*, codat în cod „2 din 4", este emis de TARP spre bus-ul *M-O*. Următoarele opt perioade sînt utilizate pentru a transmite sau manipula un cuvînt de 8 bytes în cod-mașină. În timpul celei de-a zecea perioade se emite un byte, *condition code*, de către una dintre unitățile funcționale. Acest ciclu de zece perioade este necesar din cauza organizării de tip serie-paralel a calculatorului.

O instrucțiune este executată în doi sau trei pași. În primul pas adresa instrucțiunii este trimisă de la numărătorul de locații din COP la unitățile de memorii corespunzătoare (ROM și RWM). În al doilea, unitatea de memorie adresată emite pe bus-ul *M-O* codul operației și adresa instrucțiunii tuturor unităților funcționale. Unitățile funcționale corespunzătoare recunosc codul operației, memorează adresa și inițiază execuția operației corespunzătoare. În al treilea pas se execută instrucțiunea respectivă. Primii doi pași solicită fiecare cîte un ciclu-calculator; durata celui de-al treilea pas depinde de instrucțiune și cere 0,1 cicli sau mai mult.

Setul de instrucțiuni — format din 180 instrucțiuni — include instrucțiunile aritmetice, operațiile de deplasare și logice pentru mascarea defectelor. Sînt produse și instrucțiuni pentru o serie de facilități pe buclă și subrutine. Există 28 întreruperi care pot fi mascate și testate sub controlul unui program. O clasă specială de instrucțiuni este reprezentată ce cele care facilitează toleranța la defectări: aici sînt incluse instrucțiunile de diagnosticare a defectărilor care activează mesajele de stare și logica de localizare a defectărilor din procesorul TARP, sau instrucțiunile care realizează încărcarea cu date a registrului *rollback* din unitățile procesorului TARP în modul de lucru *recovery*, ori cele care controlează alimentarea unităților de rezervă și dublarea unităților ROM sau a procesoarelor etc.

Modul de operare corespunzător unui calculator poate fi afectat de o defectare în două feluri. Astfel, un cuvînt de date sau o instrucțiune pot să fie influențate în timpul memorării, transmiterii sau procesării; efectul va fi un cuvînt-eroare. În al doilea rînd, se poate întîmpla ca în timpul execuției unei instrucțiuni un procesor sau un modul de memorie să acționeze incorect, caz în care efectul va consta într-o eroare de control. Ambele clase de erori sînt tratate în sistemul STAR prin utilizarea unor codări corespunzătoare.

Principala metodă de detecție a erorilor de control în calculatorul STAR constă în validarea faptului că fiecare unitate funcțională este activă la momentul oportun și că algoritmul corespunzător este executat în interiorul unității respective. Logica de diagnosticare a defectărilor din TARP este simplificată deoarece mesajele de stare sînt transmise procesorului TARP la anumite momente de timp fixate în interiorul fiecărui ciclu de operare. Fiecare mesaj de stare este transmis pe două fire (codat

„1 din 2"). Circuitele care formează mesajul de stare sînt dublate în fiecare unitate, pentru a face posibilă detecția unei erori în starea mesajelor.

Această combinare a dublării unităților funcționale și a unei codări redondante permite detectarea erorilor, precum și identificarea unităților defecte.

Intrarea în procesorul aritmetic principal, MAP, constă dintr-un cod al operației urmat de un operand codat, iar ieșirea este rezultatul codat urmat de un byte de condiție codat, care indică fie o neregulă (depășirea sumei sau cîtului, divizor zero), fie tipul rezultatului (pozitiv, zero sau negativ), în cazul unui rezultat bun.

Procesorul de control (COP) memorează codul de condiție și îl utilizează la implementarea instrucțiunilor pentru ramurile condiționale ale unui program. Acesta conține un numărător de locații, două registre de index și un sumator de 4 biți, care implementează indexarea adreselor codificate rezidual și incrementează numărătorul de locații.

Procesorul logic, LOP, realizează operații logice bit cu bit, precum și conversiile de cod asupra cuvintelor de intrare. Modul de operare al procesorului LOP este verificat de operarea a două copii, producîndu-se un mesaj de stare defectuoasă atunci cînd ieșirile lor diferă.

Procesorul IO/IRP primește cererile externe de întrerupere, inițiază întreruperile permise și îndeplinește o serie de funcții de intrare-ieșire.

Memoria ROM, ce conține programele permanente și constantele asociate, are o structură redondantă rezultată prin multiplicarea unității de memorie de bază.

Fiecare unitate de memorie RWM are două moduri de operare: „absolut" și „relocat". Modul de operare „relocat" produce dublarea sau triplarea memoriei pentru programe și date critice. Atunci cînd o unitate RWM, „cade" unitatea de memorie care o înlocuiește este astfel introdusă încît să fie evitată orice discontinuitate [4]. Alegerea modului de operare este realizată prin program, aceasta producînd o redondanță selectivă a memorării.

În calculatorul STAR numai procesorul LOP și memoria RWM conțin programe de sistem critice care sînt dublate. Pentru experimentare a fost prevăzut — opțional — o funcționare duplex a tuturor unităților de memorie, precum și a procesorului LOP sub controlul programului de operare.

Subsistemul principal al sistemului STAR este procesorul TARP (fig. 4.20). Acesta monitorizează funcționarea sistemului în două moduri:

— testarea informației transmise pe bus-urile de date pentru verificarea validității codării;

— testarea mesajelor de stare a unităților funcționale.

Un cuvînt incorect sau o deviere de la răspunsul așteptat al unităților funcționale va cauza întreruperea modului de lucru *standard* și va conduce sistemul în modul de lucru *recovery*.

Structura procesorului TARP este evidențiată în figura 4.22. Din punct de vedere funcțional, procesorul este divizat în două secțiuni. O secțiune controlează modul de lucru *standard* al sistemului și efectuează diagnosticarea defectărilor, iar cealaltă secțiune controlează modul de lucru *recovery* și efectuează comutarea unităților înlocuibile.

Prima secțiune, CAT (Control And Test), constă dintr-un decodor, un ceas și un numărător care generează semnalele de tact pentru modul

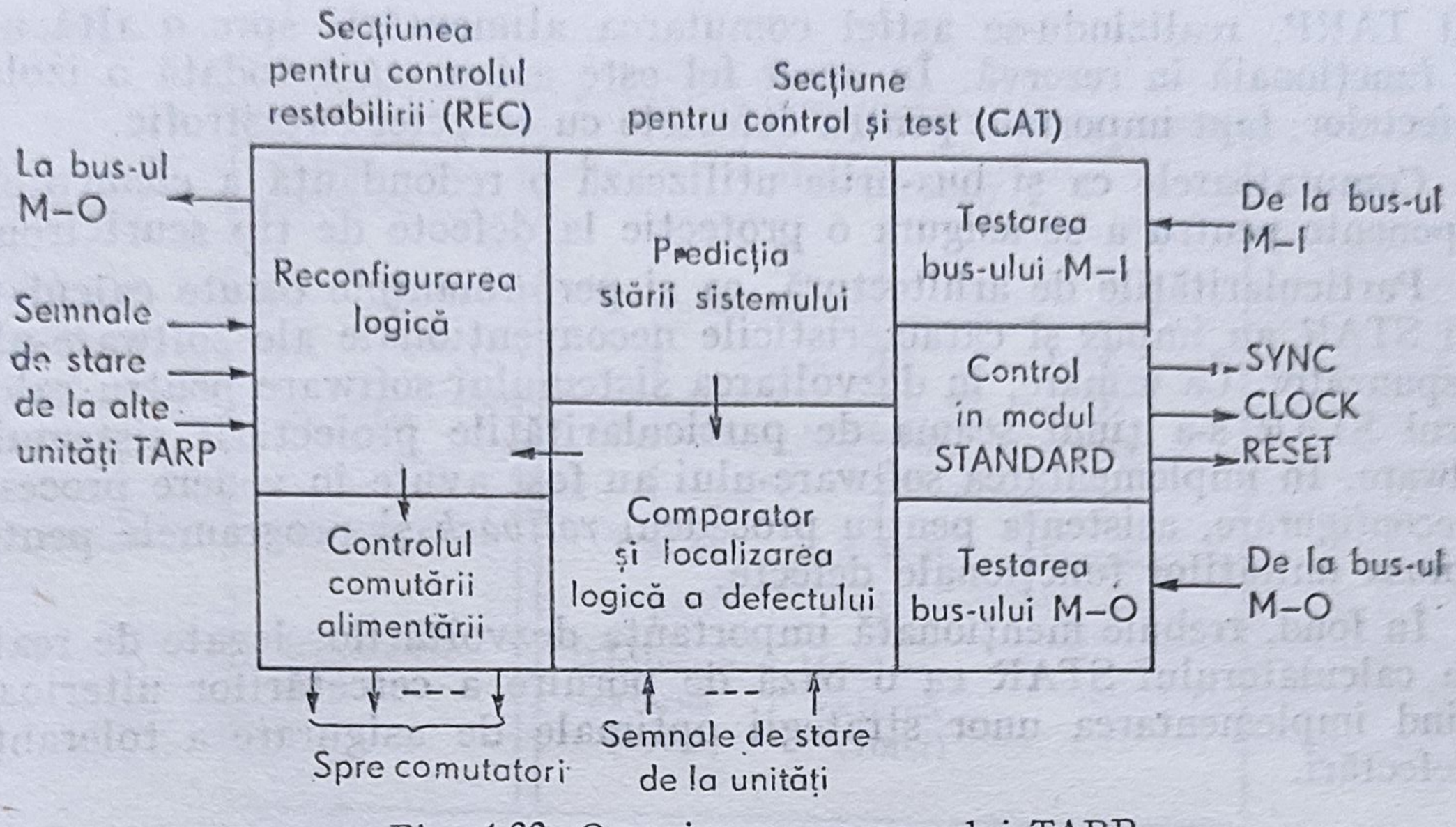


Fig. 4.22. Organizarea procesorului TARP.

de funcționare *standard*. Localizarea defectului este realizată din comparația datelor pe bus-uri și a datelor prezise, neconcordanța indicînd o stare de defect. În cazul detecției unei erori, secțiunea CAT oprește sistemul și transferă informația privitoare la eroare celei de-a doua secțiuni a blocului TARP.

Cea de-a doua secțiune, REC (REcovery Control), conține un registru de adrese *rollback point*, care localizează instrucțiunea de unde se reia operarea normală după o acțiune de restabilire.

Atunci cînd primește un mesaj de eroare de la secțiunea CAT, secțiunea REC emite semnalul RESET care aduce toate unitățile funcționale la o stare inițială; apoi se emite o instrucțiune de salt necondiționat care face ca programul să fie reluat de la adresa indicată de *rollback*. O indicație de defect repetată în aceeași unitate conduce la înlocuirea ei cu o rezervă corespunzătoare. Pentru realizarea înlocuirii, se întrerupe alimentarea unității defecte și este alimentată rezerva, care va prelua astfel funcția unității defecte.

Procesorul TARP constituie elementul de bază ce asigură toleranța la defectări a sistemului. De aceea, acesta are el însuși o structură redondantă de tip hibrid. Trei copii ale procesorului sînt alimentate — fiind astfel operante, iar n rezerve sînt în așteptare; într-o variantă a sistemului [4], $n = 2$. Cele trei procesoare TARP active lucrează într-o configurație cu logică de prag. O neconcordanță a celor trei ieșiri dictează modul de lucru *recovery*. Dacă această acțiune de restabilire a funcționării conduce la același rezultat, atunci unitatea TARP — aflată în neconcordanță — este trecută în poziția *stand-by* și o altă unitate TARP se introduce în triplet. Sistemul își reia funcția în modul de operare *standard*.

Această structură redondantă adoptată pentru procesorul TARP, care asigură performanțe de fiabilitate ridicate, necesită evident un cost ridicat. În consecință, s-a impus ca procesorul TARP să fie astfel proiectat încît complexitatea sa să fie cît mai mică.

Atunci cînd este identificat un defect la o unitate funcțională, înlocuirea unității funcționale defecte este comandată de către voterul procesо-

rului TARP, realizîndu-se astfel comutarea alimentării spre o altă unitate funcțională în rezervă. În acest fel este asigurată totodată o izolare a defectelor, fapt important pentru defectele cu caracter catastrofic.

Comutatoarele ca și bus-urile utilizează o redondanță a elementelor componente pentru a se asigura o protecție la defecte de tip scurtcircuit.

Particularitățile de arhitectură, ca și performanțele cerute calculatorului STAR au impus și caracteristicile neconvenționale ale software-ului corespunzător. Ca urmare, în dezvoltarea sistemului software pentru calculatorul STAR s-a ținut seama de particularitățile proiectării sistemului hardware. În implementarea software-ului au fost avute în vedere procesul de reconfigurare, asistența pentru procedeul *rollback* și programele pentru diagnoza unităților funcționale defecte.

În fond, trebuie menționată importanța dezvoltărilor legate de realizarea calculatorului STAR ca o bază de pornire a cercetărilor ulterioare privind implementarea unor strategii optimale de asigurare a toleranței la defectări.

4.5.2. SISTEMUL CVMP

Arhitectura calculatorului cu structură tolerantă la defectări, CVMP (*C*omputer *V*oted *M*ulti *P*rocessor) [53], dezvoltat la Universitatea Carnegie Mellon este evidențiată în figura 4.23. În principal, sistemul CVMP este format din trei procesoare, trei unități de memorie și un voter. Sistemul analizat poate funcționa în trei moduri de operare și anume:

- modul *independent*: trei cupluri procesor-memorie care lucrează pe sarcini independente;
- modul *difuzie*: acest mod permite difuzia informației pornind de la o sursă unică (procesor sau memorie) spre toate unitățile receptoare (memorii sau procesoare);
- modul *toleranță la defectări*: este modul de operare care interesează aici; toate unitățile lucrează pe o sarcină unică, votarea fiind realizată la fiecare transfer al informației între memorie și procesor.

Funcțional — în modul considerat — acest calculator este un monoprocesor. Strategiile de implementare a toleranței la defectări utilizate sînt indicate în figura 4.24. De menționat că a fost prevăzută posibilitatea triplării voterului; în acest caz partiția procesor, memorie, voter nu va cuprinde decît un ansamblu — calculatorul.

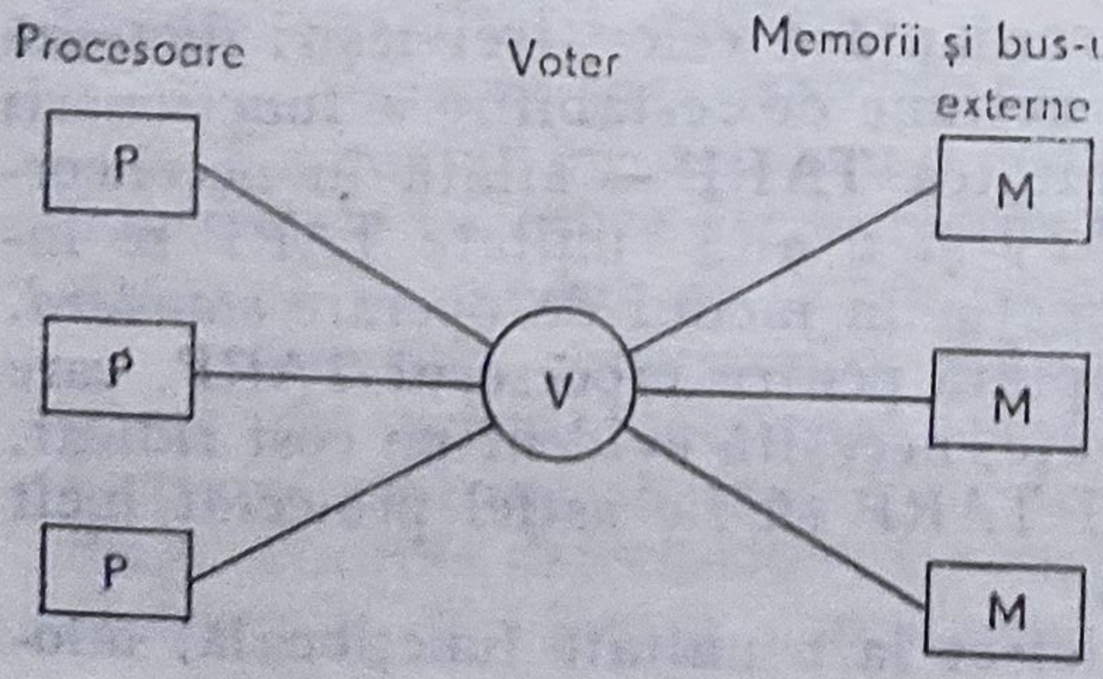


Fig. 4.23. Arhitectura sistemului CVMP.

Voterul este bidirecțional: de aceea, din punct de vedere funcțional, se consideră că există două votere — unul în aval de procesoare, iar celălalt în aval de memorii.

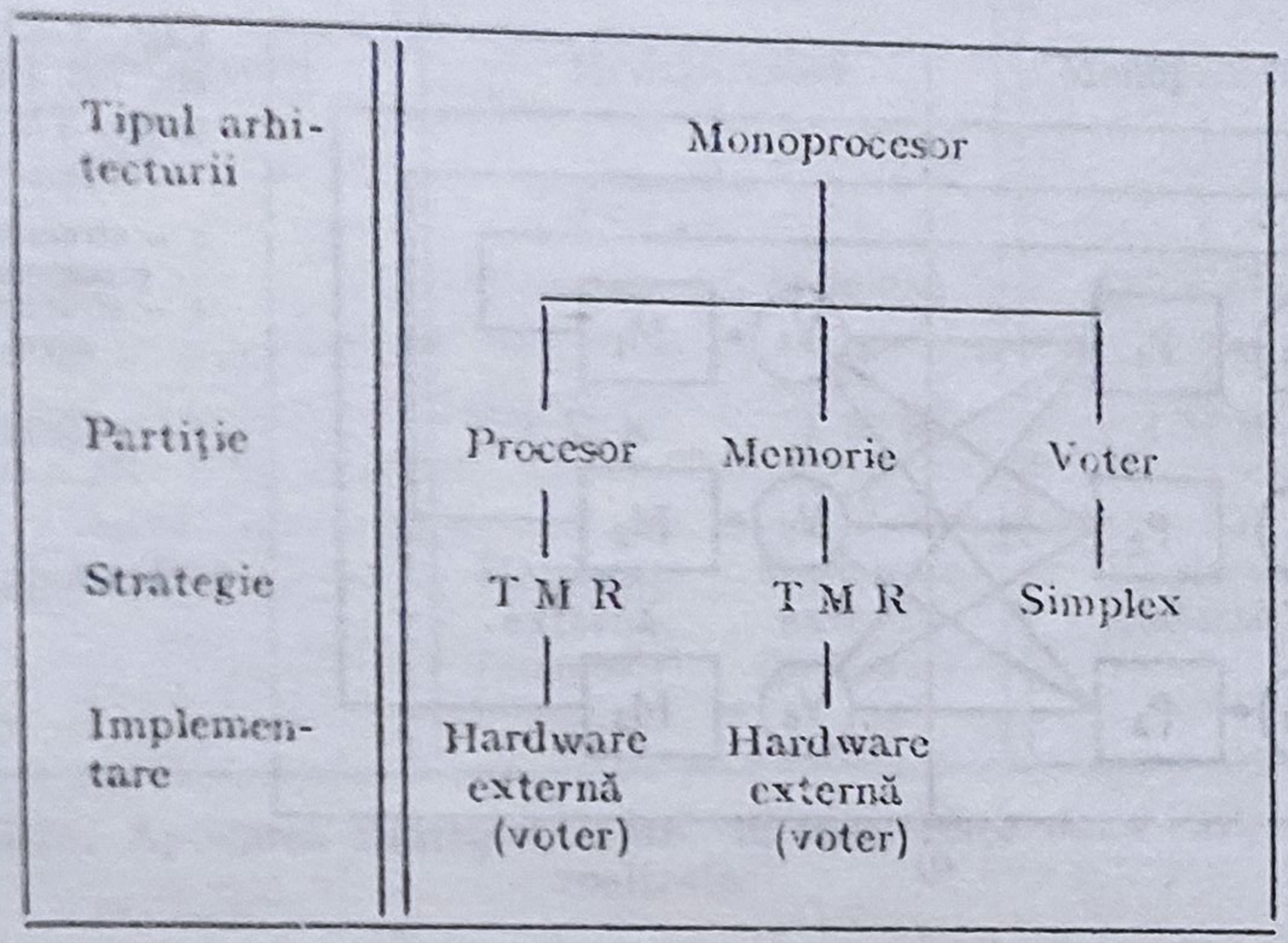


Fig. 4.24. Aplicarea procedeelor pentru tolerarea defectărilor în sistemul CVMP.

4.5.3. MICROCALCULATOR CU STRUCTURĂ TMR

Într-un studiu [60] asupra utilizării strategiei TMR în cazul microcalculatoarelor se analizează două tipuri de arhitecturi tolerante la defectări, prezentate în figurile 4.25 *a* și *b*. Se adoptă arhitectura din figura 4.25 *b* care minimalizează structura și numărul voterelor.

În figura 4.26 sînt indicate sintetic caracteristicile de tolerare la defectări pentru cele două arhitecturi menționate.

Se remarcă faptul că structura calculatorului CVMP, cu trei votere bidirecționale, se apropie funcțional de primul tip de arhitectură (fig. 4.25 *a*) — — voter implementat hardware în aval de procesoare și în aval de memorii, dar din punctul de vedere al influenței defectărilor însă este mai apropiat de cel de-al doilea tip de arhitectură considerat (fig. 4.25 *b*). De menționat că un defect într-un voter bidirecțional induce o funcționare defectuoasă atît într-o memorie, cît și într-un procesor.

4.5.4. SISTEMUL COPRA

Sistemul tolerant de defectări COPRA (Calculateur à Organisation Parallèle Reconfigurable Automatiquement) a fost realizat [45] de către SAGEM (Société d'Applications Générales d'Electronique et de Mecanique) și EMD (Electronique Marcel Dessault). Funcțional, sistemul COPRA este un multiprocesor, în care numărul unităților poate, apriori, — varia. În continuare se va analiza versiunea de bază a acestui sistem care, din punct de vedere funcțional, apare ca un biprocesor.

Toleranța la defectări a sistemului COPRA se realizează utilizîndu-se următoarele principii:

— strategiile de detecție a erorilor și înlocuire a unităților defecte sînt implementate prin mijloace hardware (procesoare dublate, utilizarea codurilor redondante la nivelul memoriilor etc.);

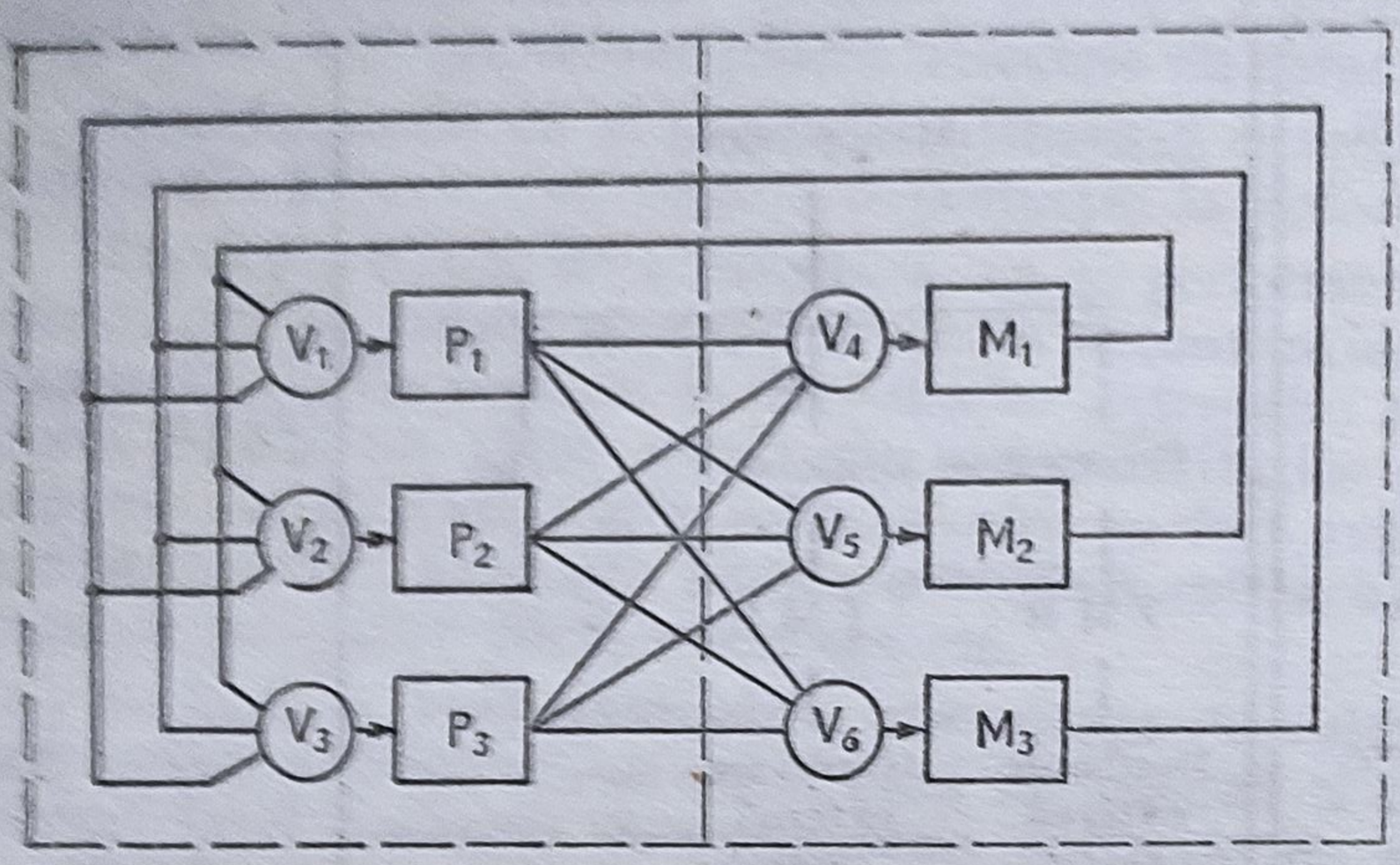
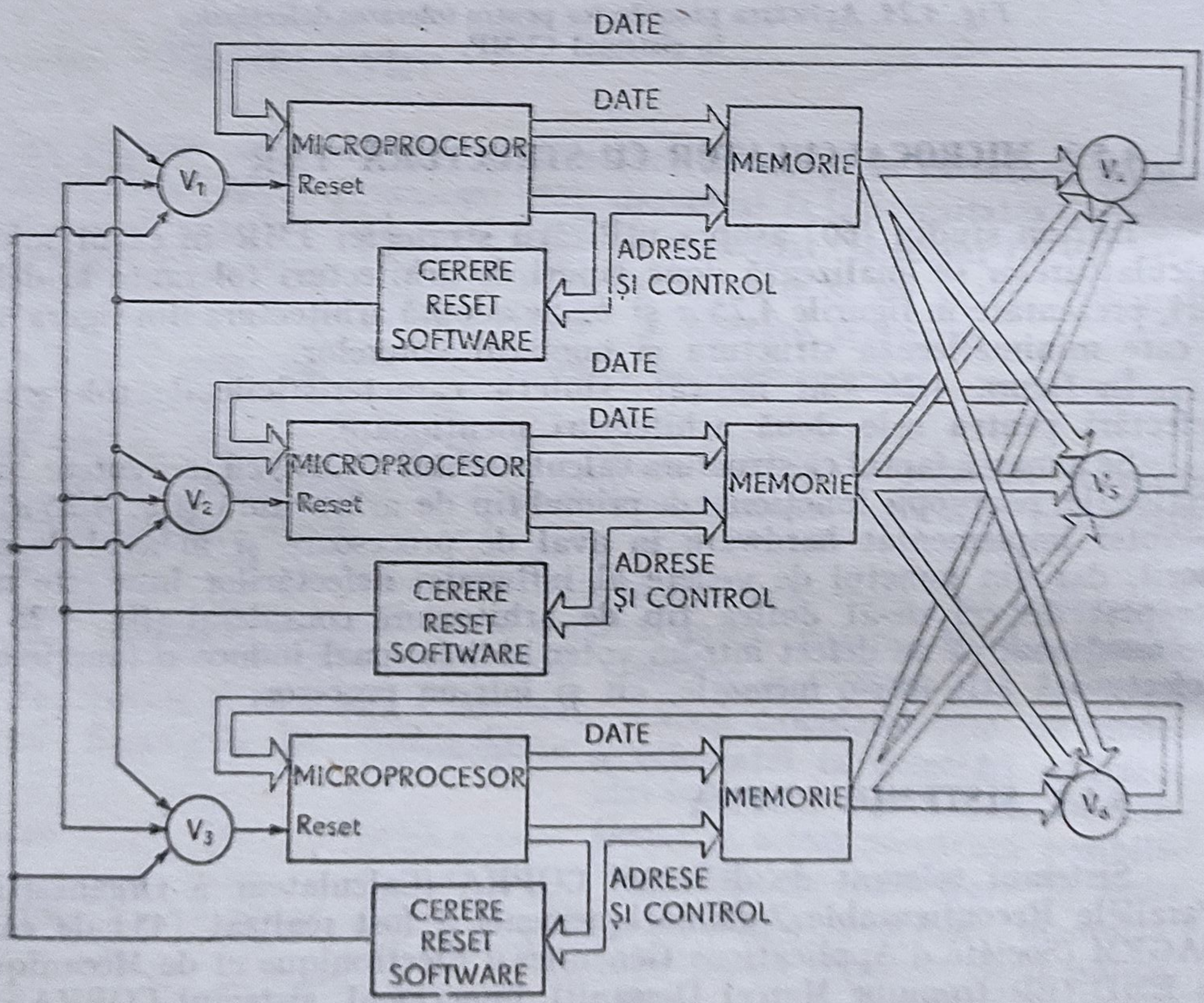


Fig. 4.25. Tipuri de arhitecturi care folosesc strategia TMR pentru tolerarea defectărilor:

a — structura TMR aplicată procesorului și memoriei;
b — structura TMR aplicată la nivel de calculator.

a)



b)

— folosirea testelor periodice pentru realizarea validității elementelor critice și, în primul rînd, a elementelor hardware cu rol important în asigurarea toleranței la defectări.

Fiecare sarcină este asigurată de către două procesoare în regim duplex pe o unitate de memorie, iar detecția erorilor se realizează prin utilizarea codurilor de paritate; o altă „copie" a memoriei se află în rezervă pentru

Tipul arhitecturii	Monoprocesor	Monoprocesor
	Procesor / Memorie	Calculator
Partiție		
Strategie	T M R / T M R	T M R
Implementare	Hardware externă (memorie) / Hardware externă (procesor)	Hardware internă

Fig. 4.26. Aplicarea strategiei TMR în cazul celor două arhitecturi analizate.

a intra în sistemul restabilirilor. La apariția unei defectări permanente aceste unități — un „duplex" sau unitatea de memorie — sînt eliminate, sarcinile mai puțin critice fiind abandonate, iar misiunea sistemului este reluată de unitățile valide — al doilea duplex sau a doua unitate de memorie [48]. În figura 4.27 se prezintă arhitectura sistemului COPRA.

La nivelul modulelor de comunicații — pentru a face posibilă acoperirea defectelor alocatorului — fiecare procesor posedă un subsistem de avertizare de tip *watch-dog* cu rolul de a evita blocarea sa abuzivă în cursul unui schimb de informație cu o unitate de memorie. În plus, pentru identificarea stărilor eronate, atît modulele *watch-dog*, cît și alocatoarele sînt testate periodic.

Legat de arhitectura sistemului (fig. 4.27) se mai remarcă următoarele:

• datele prelucrate de procesoare sînt comparate prin intermediul unui comparator asociat fiecărei perechi de procesoare, în timp ce comenzile memoriei elaborate de aceste procesoare sînt comparate de comparatoarele asociate fiecărui modul de memorie;

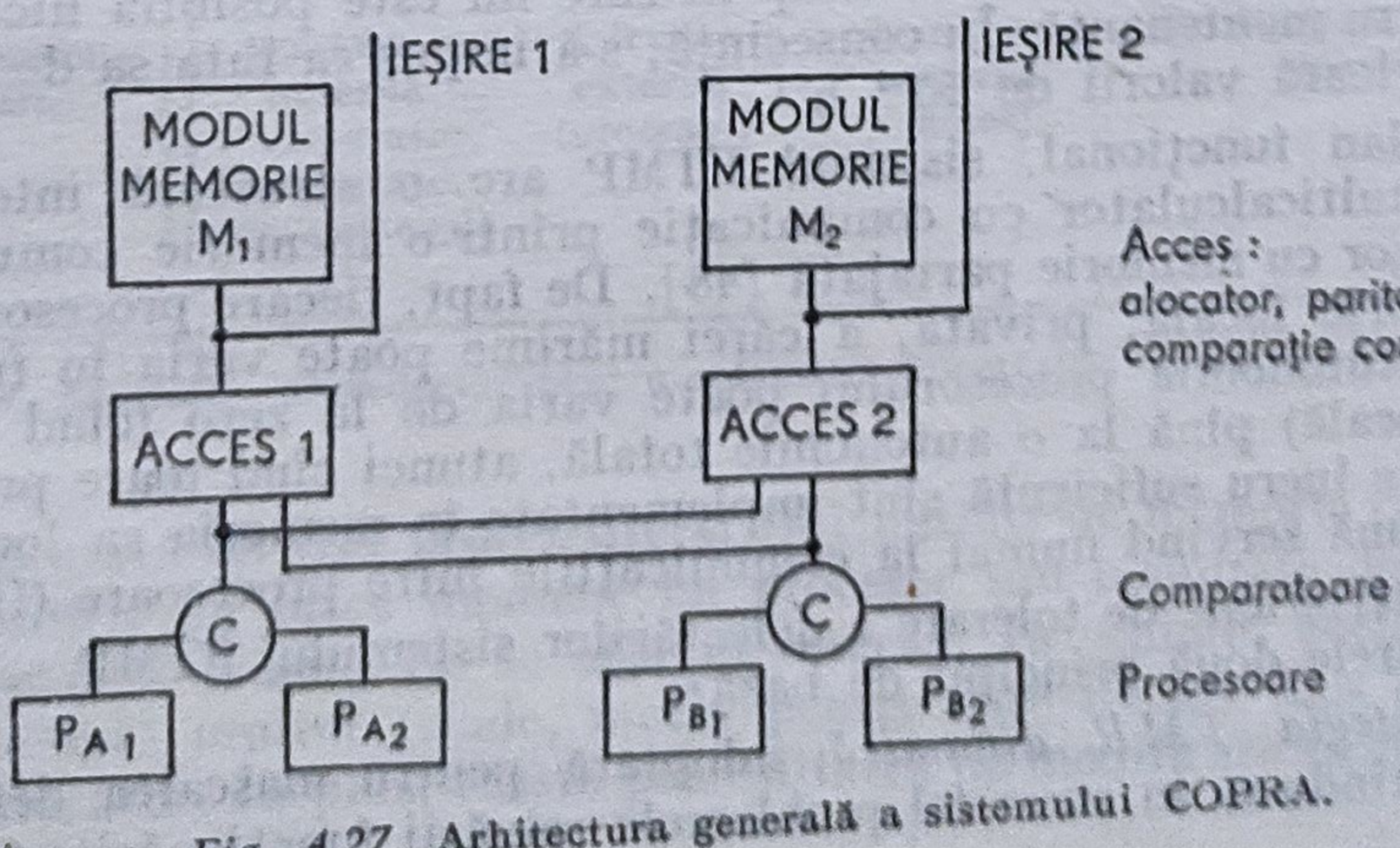


Fig. 4.27. Arhitectura generală a sistemului COPRA.

• comparatoarele de date ca şi circuitele modulului *watch-dog* urmează aceeaşi strategie de identificare a defectărilor (teste periodice) şi înlocuire (ele sînt asociate funcţional unei perechi de procesoare); orice defectare a unuia dintre aceste elemente ale sistemului antrenează înlocuirea perechii de procesoare prin alte unităţi similare;

• alocatoarele de memorie, detectoarele de paritate ca şi comparatoarele de comandă urmează o strategie de detecţie a defectărilor similară: testări periodice, utilizarea unor sisteme de tip *watch-dog* şi înlocuire: sistemele *watch-dog* avertizează procesorul atît în caz de defectare a alocatorului, provocînd blocarea sa, cît şi în cazul blocării celeilalte perechi de procesoare, acest incident fiind imputabil unei funcţionări defectuoase a comparatorului de comandă. Se menţionează că înlocuirile sînt asociate funcţional unui modul de memorie, astfel încît orice defectare atrage după sine atît înlocuirea modulului, cît şi a altor elemente.

Fiabilitatea sursei de alimentare este mărită prin utilizarea a două blocuri de alimentare conectate într-o structură de tip *quadding* [48].

Implantarea materială a acestor structuri impune o sincronizare corespunzătoare între unităţile componente. Ca urmare, în scopul creşterii fiabilităţii s-a adoptat o structură redondantă logică majoritară de tip „2 din 3", utilizîndu-se trei generatoare de tact, punctul critic al sistemului fiind voterul acestei structuri.

În figura 4.28 sînt prezentate sintetic strategiile utilizate pentru realizarea toleranţei la defectări în sistemul COPRA. Se remarcă faptul, subliniat şi în paragrafele anterioare, că problemele reluării funcţionării reprezintă punctul cheie pentru validitatea strategiilor detecţie-înlocuire în diferitele unităţi ale sistemului COPRA. Realizarea sistemului COPRA a condus la rezultate semnificative în acest domeniu [45].

4.5.5. SISTEMUL FTMP

Sistemul FTMP (Fault Tolerant Multiprocessor) — calculator tolerant la defectări realizat la Charles Strack Draper Laboratory [32] — este destinat aplicaţiilor aeronautice şi spaţiale de înaltă fiabilitate pentru o durată a misiunii de cel puţin 10 ore, timp în care nu este posibilă nici o intervenţie pentru mentenanţă. În consecinţă, s-a impus ca rata sa de defectare să fie inferioară valorii de 10^{-9} h^{-1}.

Pa plan funcţional, sistemul FTMP are o structură intermediară între un multicalculator cu comunicaţie printr-o memorie comună şi un multiprocesor cu memorie partajată [48]. De fapt, fiecare procesor dispune de o memorie locală, privată, a cărei mărime poate varia în funcţie de aplicaţie. Autonomia procesorului poate varia de la zero (cînd nu are o memorie locală) pînă la o autonomie totală, atunci cînd toate programele şi o zonă de lucru suficientă sînt implementate în memoria sa locală, memoria comună servind numai la comunicaţiile între procesoare (fig. 4.29).

Caracteristicile de tolerare a defectărilor sistemului FTMP se bazează pe următoarele două principii de bază;

• *Strategia TMR dinamică*, adoptată pentru mascarea defectărilor simple, evitîndu-se în acest fel problemele reluării funcţionării. O sarcină

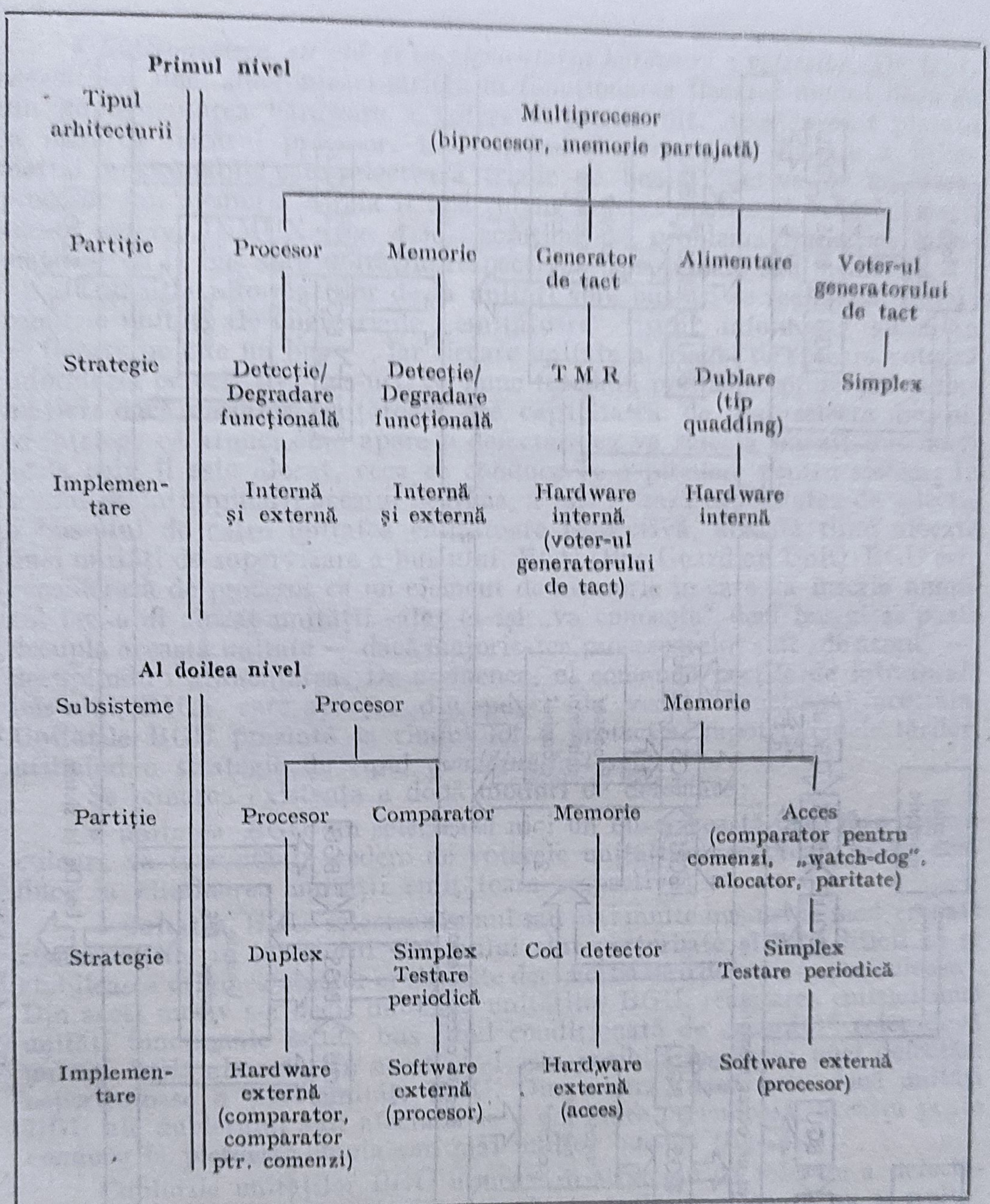


Fig. 4.28. Aplicarea procedeelor pentru tolerarea defectărilor în cazul sistemului COPRA.

este executată de trei procesoare şi trei unităţi de memorie conectate prin trei bus-uri. La orice moment un procesor sau o unitate de memorie fie aparţine unei triade, fie se află în rezervă. În schimb, acelaşi bus poate aparţine mai multor triade, motiv pentru care fiecare bus este prevăzut cu un alocator. În plus, există posibilitatea de modificare în timp a acestor triade.

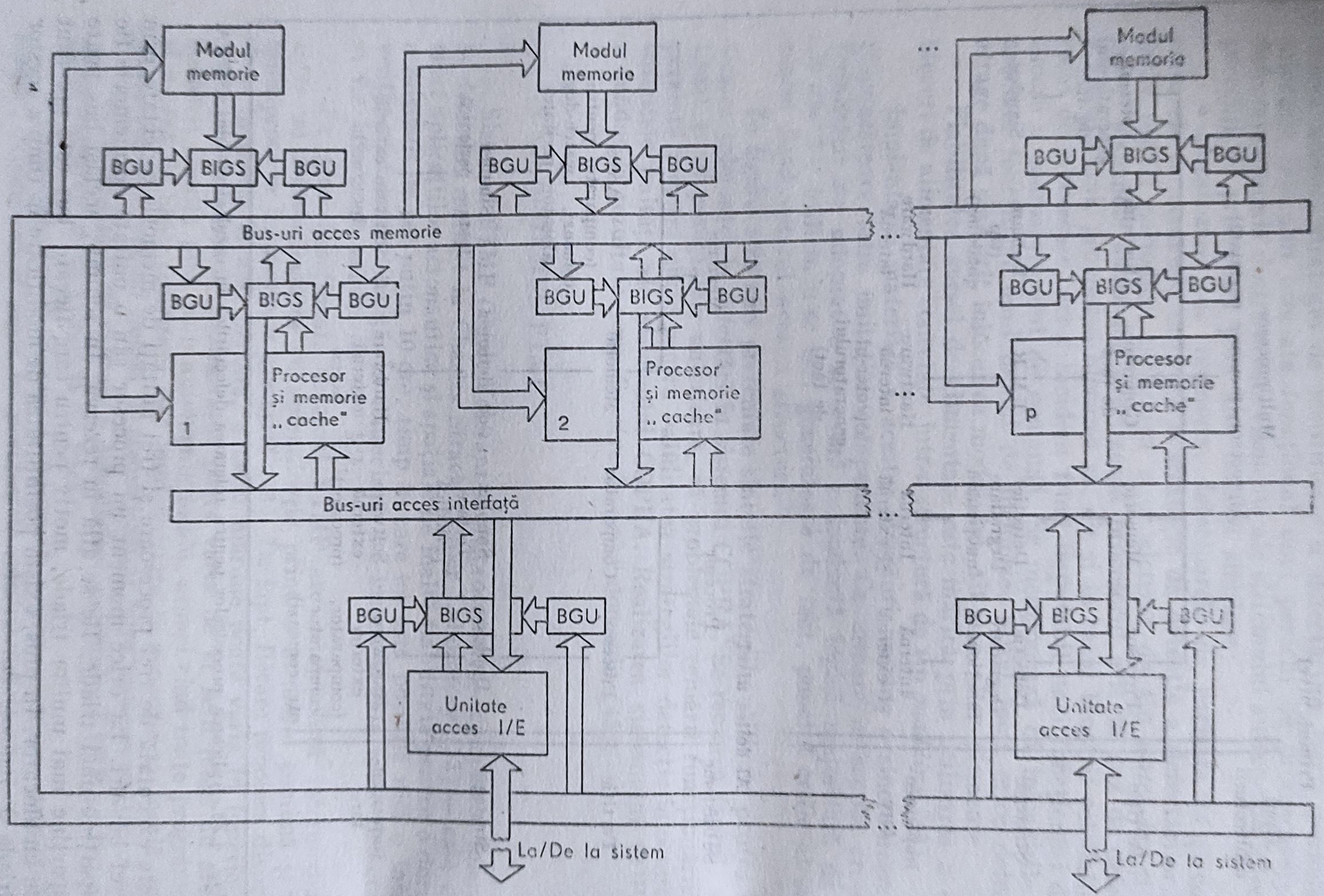


Fig. 4.29. Arhitectura generală a sistemului FTMP.

• *Sincronizarea strictă și implementarea hardware a voterelor*. De fapt, necesitatea unei sincronizări stricte în funcționarea fiecărui modul decurge din implementarea hardware a voterelor bit cu bit. Acestea sînt plasate la intrarea fiecărui procesor. Ele cuprind și circuite de dirijare a informației programabile care selectează triade de bus-uri active cu unitatea-procesor sau memoria căreia îi aparțin în vederea realizării votării. Deci, aceste votere, INMUX (fig. 4.30), soluționează problema tranziției informațiilor de la bus spre unitățile respective — procesoare sau memorii.

Tranziția informațiilor de la unități spre bus-uri se realizează astfel: cele trei unități ale unei triade „emițătoare" trimit informația simultan — fiecare pe cîte un bus —, iar fiecare unitate a triadei receptoare votează informația pe cele trei bus-uri. Se pune totodată problema protecției informației: dacă unitatea emițătoare are capacitatea de a-și selecta bus-ul, se înțelege că atunci cînd apare o defectare ea va selecta un alt bus decît acela care îi este alocat, ceea ce conduce la o pierdere pentru sistem. În scopul preîntîmpinării acestui neajuns, a fost retrasă capacitatea de selecție a bus-ului de către unitatea emițătoare respectivă, aceasta fiind alocată unei unități de supervizare a bus-ului, BGU (Bus Guardian Unit). BGU este considerată de procesor ca un element de memorie în care va înscrie numărul bus-ului alocat unității sale; el își „va cunoaște" deci bus-ul și poate decupla această unitate — dacă majoritatea procesoarelor sînt „de acord" —, decuplîndu-i alimentarea. De asemenea, el comandă porțile de intrare ale bus-ului, BIGS, care aparțin din punct de vedere funcțional acestuia. Unitățile BGU prezintă la rîndul lor o protecție împotriva defectărilor, utilizînd o strategie de tipul *quadding-fail safe*.

Se remarcă existența a două moduri de defectare:

• unitatea BGU nu selectează nici un bus: această defectare nepericuloasă va fi avută în vedere de voterele unităților receptoare și va conduce la eliminarea unității emițătoare respective;

• unitatea BGU selectează unul sau mai multe bus-uri în mod eronat: comunicațiile în interiorul sistemului sînt perturbate și este dificil să se stabilească originea acestor erori; este deci vorba de o defectare „periculoasă". Din acest motiv s-a decis dublarea unităților BGU, realizarea emisiei unei unități funcționale pe un bus fiind condiționată de „acordul" celor două unități BGU ale sale. În acest mod este posibilă mascarea unei defectări nepericuloase a unei unități BGU. Din contra, dacă cele două unități BGU ale dubletului sînt afectate de o defectare periculoasă, aceasta poate conduce la pierderea unuia sau mai multor bus-uri (fig. 4.31).

Cuplurile unităților BGU urmează o strategie de tolerare a defectărilor de tip *quadding*, cu mențiunea că strategia menționată nu se aplică decît pentru modul lor de defectare de tip „periculos".

Necesitatea unei sincronizări perfecte în funcționarea unităților componente implică utilizarea unui generator de tact central cu structură tolerantă la defectări. Structura adoptată pentru acest subsistem [32] este prezentată în figura 4.32.

Pentru o evidențiere mai clară a caracteristicilor de tolerare a defectărilor este utilă descrierea funcționării sistemului în termenii specifici calculatoarelor sale componente, constituite din triade de procesoare, bus-uri și memorii. Configurația sistemului la un moment dat este determinată

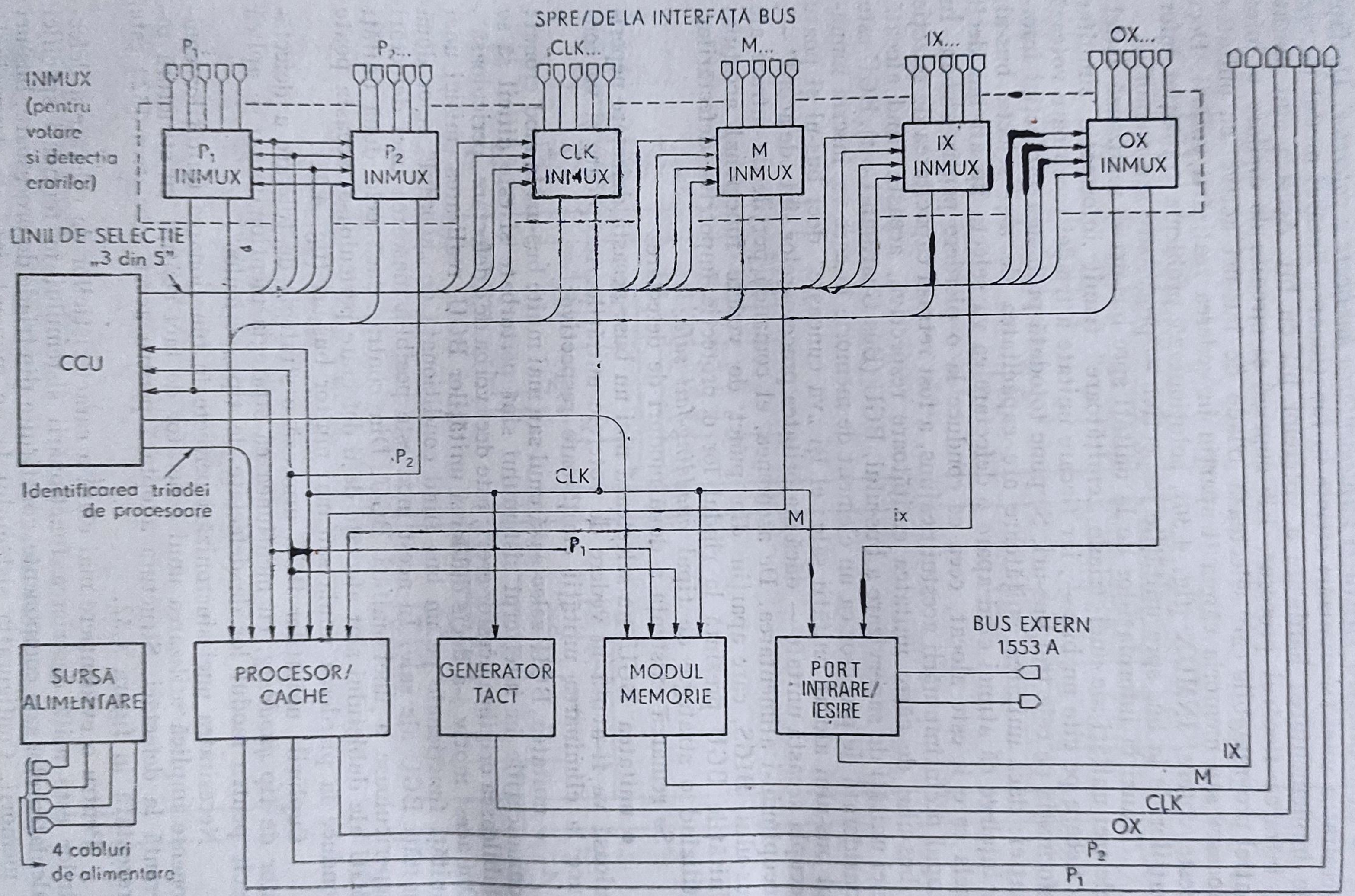


Fig. 4.30. **Structura voterelor în sistemul FTMP.**

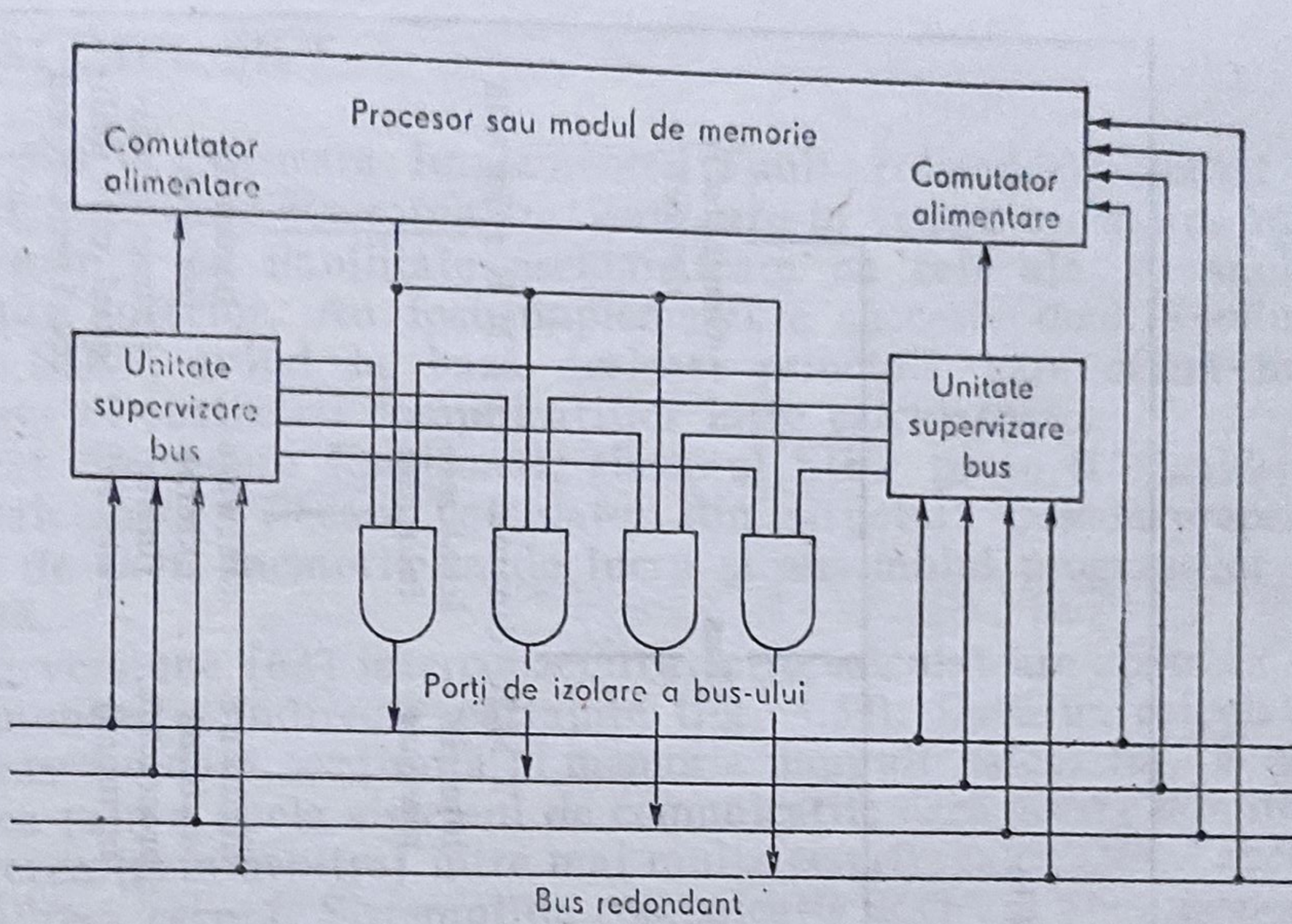


Fig. 4.31. Conectarea modulelor BGU în sistemul FTMP.

de conținutul memoriilor, voterelor, care vor determina bus-urile pe care se va vota, precum și de unitățile BGU ce determină bus-urile pe care se va emite informația. Calculatoarele pot modifica conținutul acestor memorii. În plus, dezacordurile apărute în cursul votărilor sînt memorate de către votere și sînt luate în considerație de către calculatoare.

În figura 4.33 sînt prezentate sintetic strategiile avute în vedere pentru obținerea caracteristicilor de toleranță la defectări ale acestui sistem.

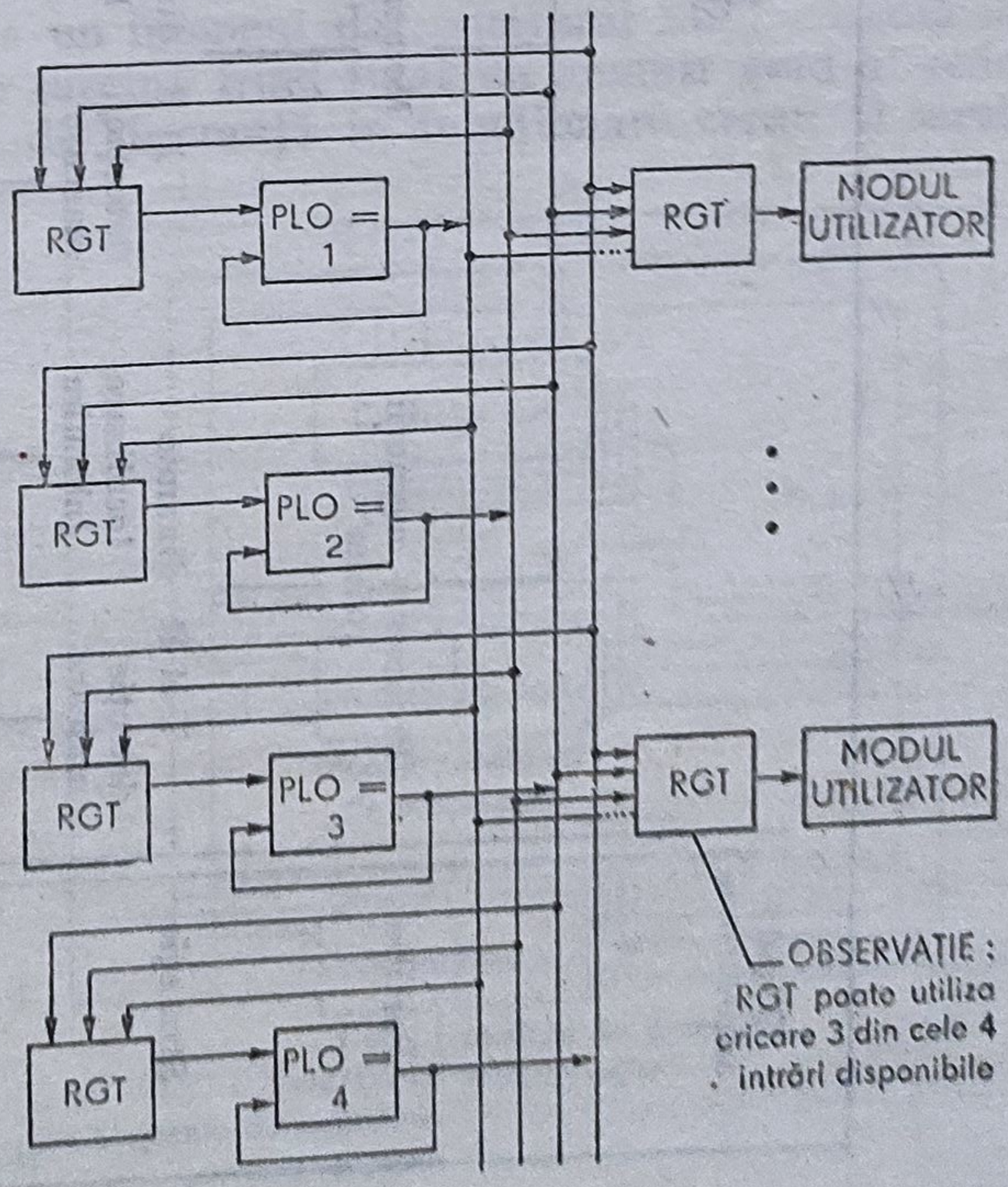


Fig. 4.32. Structura generatorului de tact în sistemul FTMP:
RGT — receptor generator tact; PLO — — oscilator *phase-locked*.

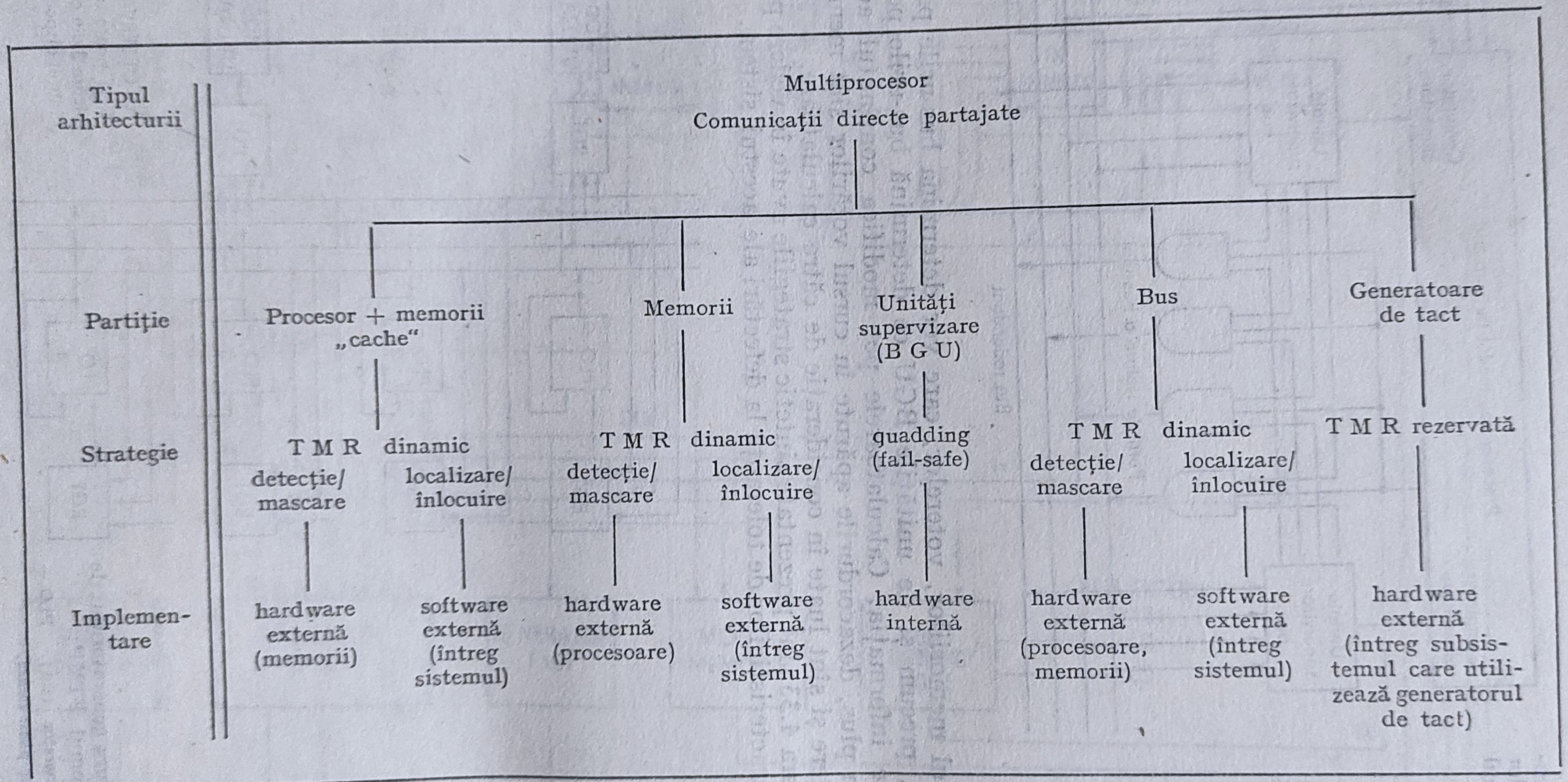


Fig. 4.33. Aplicarea procedeelor pentru tolerarea defectărilor în cazul sistemului FTMP.

4.5.6. SISTEMUL SIFT

Sistemul SIFT (Software Implemented Fault Tolerance), studiat și implementat la Standford Research Institute, are în vedere o clasă de aplicații și performanțe de fiabilitate asemănătoare cu cele ale sistemului FTMP prezentat anterior. Au fost implementate succesiv două versiuni ale sistemului SIFT, avînd la bază aceleași principii, care diferă însă între ele la nivelul realizării comunicațiilor între calculatoare.

Din punct de vedere funcțional, sistemul SIFT poate fi considerat ca un multicalculator; fiecare calculator din structură posedă propriul său generator de tact, memoria sa de lucru și ansamblul programelor pe care le execută.

În prima versiune [62] interconectarea între calculatoare apare ca un sistem de comunicații indirecte partajate (fig. 4.34). Dacă un calculator dorește să citească o dată conținută în memoria unui alt calculator, acesta începe acțiunea prin a apela sistemul de comunicații; dacă acesta este liber și acceptă cererea (prin arbitraj între mai multe cereri), calculatorul apelat îi transmite adresa cerută. Sistemul de comunicație apelează apoi memoria dorită și, îndată ce o obține, transmite data cerută.

În a doua versiune [61] sistemul SIFT prezintă interconectările între calculatoarele componente ca un sistem de comunicații directe, specificate (unidirecționale) (fig. 4.35). Un calculator emite printr-un emițător-serie — — care îi este propriu — spre celelalte calculatoare receptoare; fiecare dintre acestea primește printr-un modul — care îi este propriu de asemenea toate emisiunile provenind de la alte calculatoare.

Principiul adoptat în cazul sistemului SIFT pentru realizarea toleranței la defectări constă în folosirea redondanței NMR dinamice implementată pe cale software. La un moment dat, sistemul SIFT execută mai multe sarcini, pentru fiecare sarcină fiind fixat un anumit grad al redondanței în funcție de gradul de siguranță în funcționare cerut. O sarcină

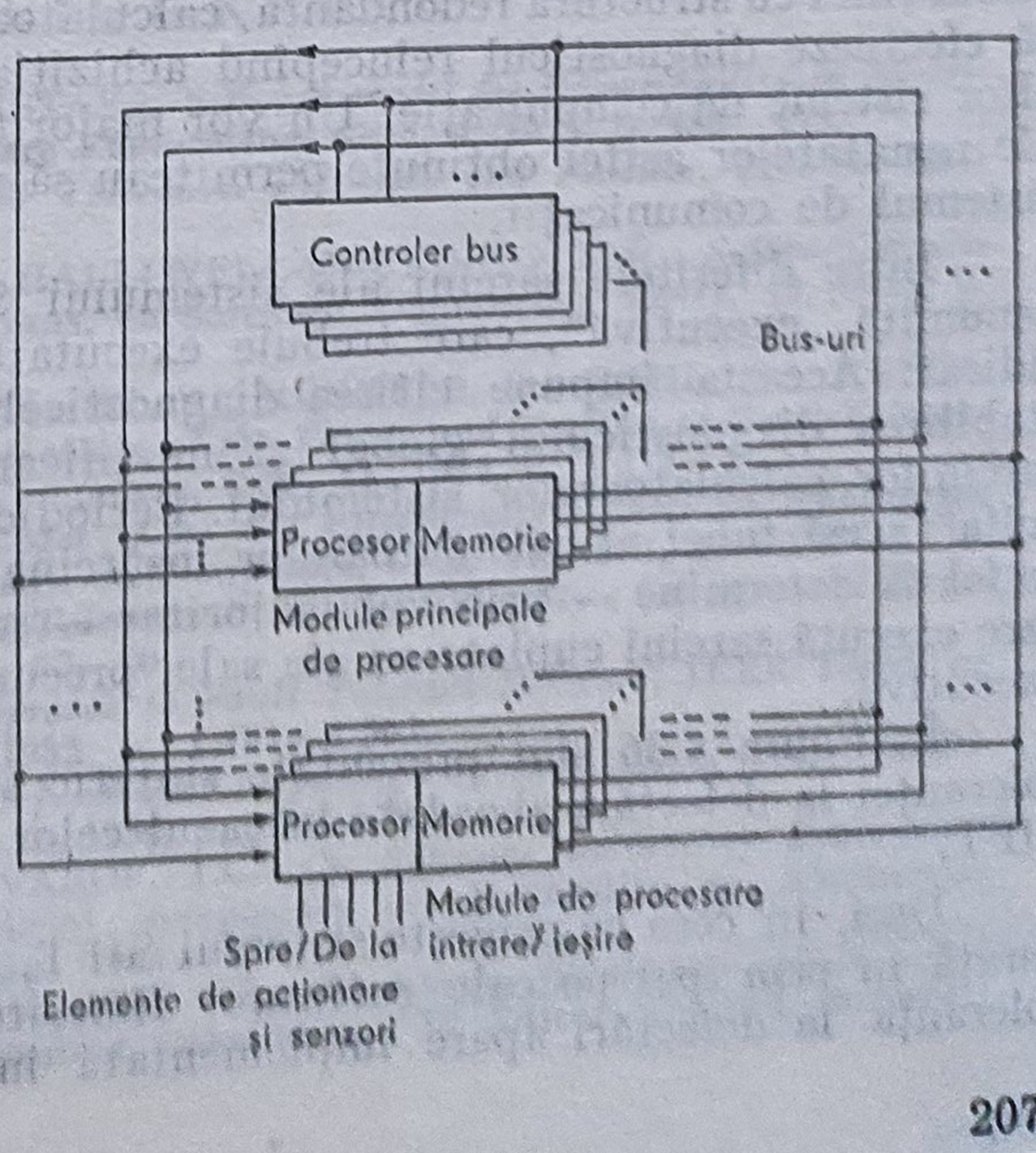


Fig. 4.34. Structura generală a primei versiuni a sistemului SIFT.

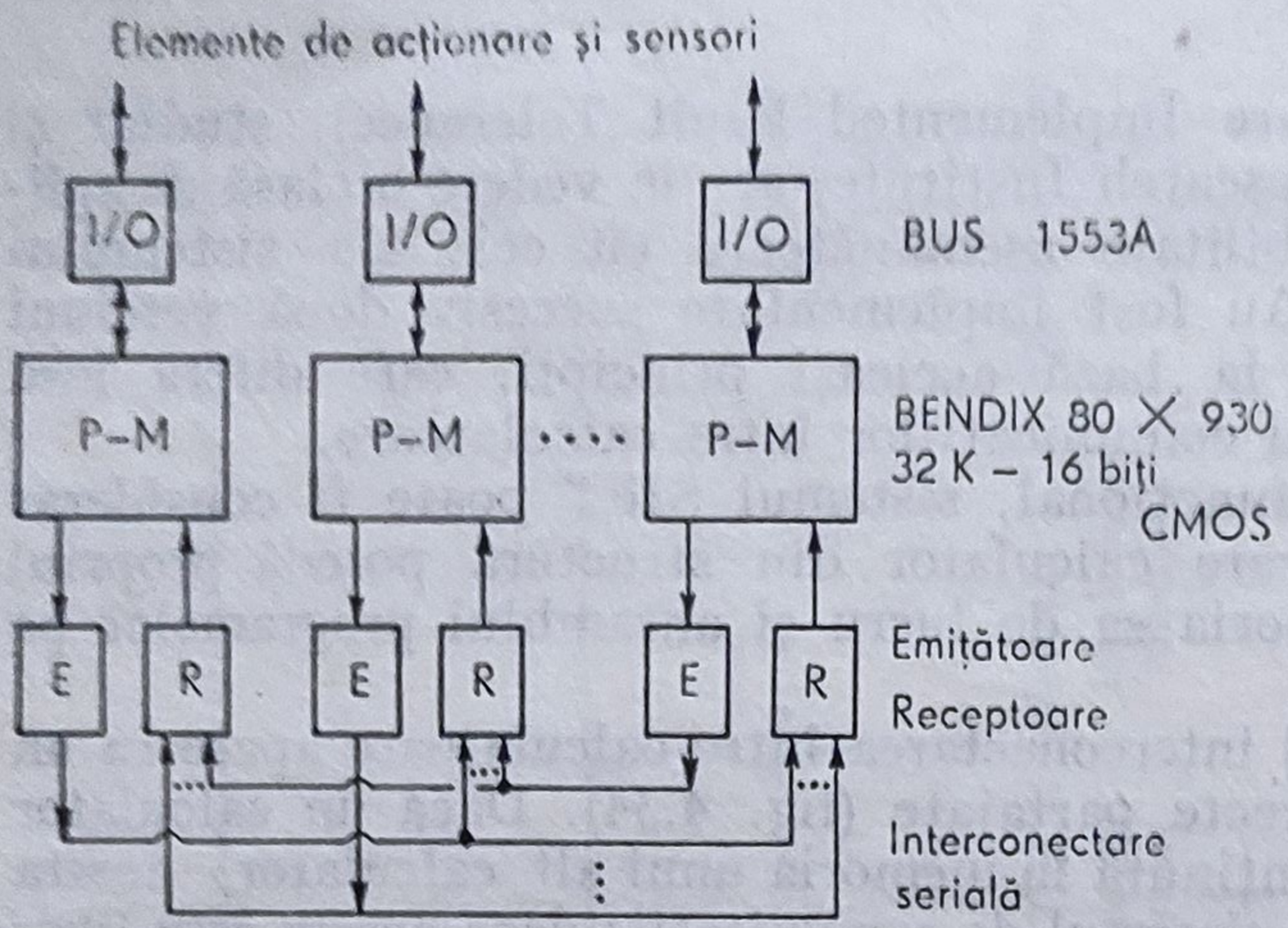


Fig. 4.35. Structura generală a celei de a doua versiuni a sistemului SIFT:

I/O – module de procesare intrare-ieșire; P-M – module principale de procesare; E – emițător; R – receptor.

care cere gradul redondanței n, este executată de n calculatoare cvasisimultan; în fapt, aceste calculatoare nu sînt sincrone, avînd fiecare propriul generator de tact; astfel un calculator poate participa la mai multe grupe de n calculatoare corespunzătoare realizării mai multor sarcini.

Dacă un calculator trebuie să achiziționeze o dată corespunzătoare unei alte sarcini, atunci el va achiziționa datele tuturor calculatoarelor care execută această sarcină și votează asupra acestei date. În a doua versiune a sistemului, în cazul apariției unui dezacord în cursul acestei votări, toate calculatoarele care au achiziționat data respectivă sînt capabile să diagnosticheze care unitate emițătoare este defectă. În cazul primei versiuni a sistemului, problema diagnosticării se pune doar în raport cu sistemul de comunicație; ea se referă la determinarea provenienței erorii de la un calculator sau de la un element al sistemului de comunicație. Sistemul fiind cu structură redondantă, calculatoarele care primeau data puteau să efectueze diagnosticul reîncepînd achiziția acestei date prin utilizarea altor sisteme de comunicație. Un vot majoritar asupra diferitelor versiuni ale rezultatelor astfel obținute permiteau să se pună — sau nu — în cauză sistemul de comunicații.

Între diferitele sarcini ale sistemului SIFT există una particulară, denumită „executivă", care trebuie executată cu un grad de redondanță ridicat. Aceasta impune citirea diagnosticelor elaborate de calculatoare, stabilirea diagnosticului global și modificarea tabelului de alocare a sarcinilor calculatoarelor sistemului. Periodic, fiecare calculator va „consulta" acest tabel al calculatoarelor însărcinate ca executive. El va putea astfel să determine — prin vot majoritar — sarcina de execuție, partenerii care execută sarcini cuplate cu ale sale, precum și calculatoarele cu sarcini executive.

În figura 4.36 sînt prezentate sintetic strategiile de implementare a toleranței la defectări abordate în cazul celor două versiuni ale sistemului SIFT.

Deci, în ceea ce privește sistemul SIFT, toleranța la defectări este asigurată în principal pe cale software. Referitor la arhitectura materială, toleranța la defectări apare implementată în două moduri:

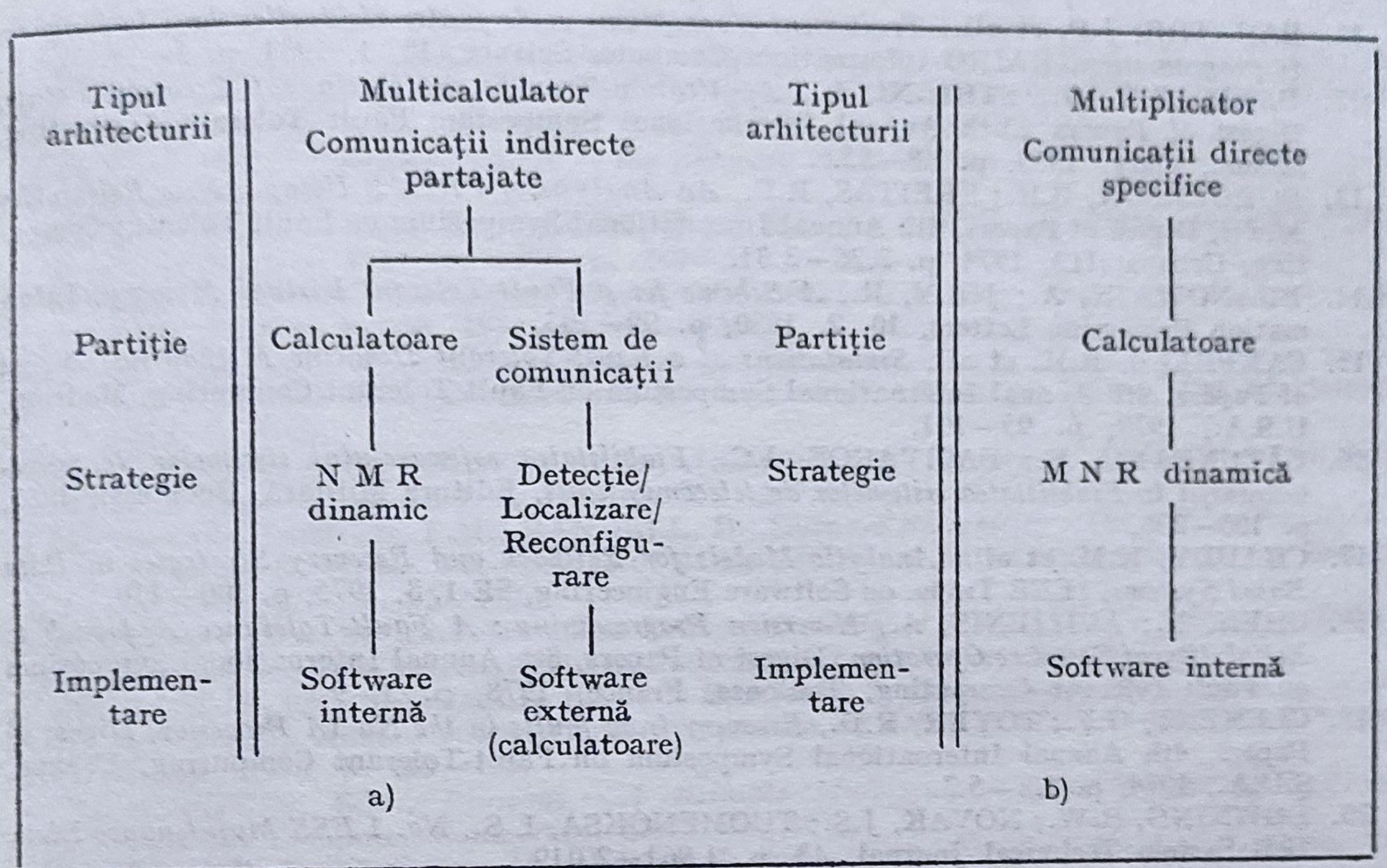


Fig. 4.36. Aplicarea procedeelor de tolerare a defectărilor în cazul sistemului SIFT:
a — în cazul primei versiuni; *b* — în cazul celei de a doua versiuni.

— redondanța elementelor la nivelul fiecărui subsistem;
— izolarea completă din punctul de vedere al defectărilor între fiecare element al unui subsistem, precum și între subsistemele distincte ale sistemului.

BIBLIOGRAFIE

1. ANDERSON, T.; LEE, P.A.; SHRISTAVA, S.K., *A model for Recoverability in Multi-level Systems*, IEEE Trans. on Software Engineering, **SE-4**, 6, 1978, p. 486—493.
2. ANDERSON, T.; LEE, P.A., *Fault Tolerance. Principle and Practice*, Prentice-Hall International Inc., London, 1981.
3. ANDERSON, T.; BARRET, P.A.; HALLIWELL-MOULDING, M.R., *Softwars Fault-Tolerance : An Evaluation*, IEEE Trans. on Software Engineering, **SE-11**, 12, 1985, p. 1 502—1 510.
4. AVIZIENIS, A.; GILLEY, G.; MATHUR, F.; RENNELS, D.; ROHR, J.; RUBIN, D., *The STAR Computer : An Investigation of the Theory and Practice of Fault-Tolerant Computer Design*, IEEE Trans. on Computers, **C-20**, 11, 1971, p. 1 312—1 321.
5. AVIZIENIS, A., *Arithmetic Error Codes ; Cost and Effectiveness Studies for Application in Digital System Design*, IEEE Trans. on Computers, **C-20**, 11, 1971, p. 1 322—1 331.
6. AVIZIENIS, A.; KELLY, J., *Fault Tolerance by Design Diversity : Concepts and Experiments*, Computer, **17**, 8, 1984, p. 67—80.
7. AVIZIENIS, A., *The N-version Approach to Fault Tolerant Software*, IEEE Trans. on Software Engineering, **SE-11**, 12, 1985, p. 1 491—1 501.
8. AVIZIENIS, A.; LAPRIE, J.C., *Dependable Computing : From Concepts to Design Diversity*, Proceedings of the IEEE, 74, 5, 1986, p. 629—638.
9. BACIVAROF, ANGELICA; BACIVAROF, I.C., *On Software Reliability Evaluation*, Proceedings of the 4-th International Symposium on Reliability in Electronics, RELECTRONIC, Budapest, Hungary, 1981.
10. BACIVAROF, ANGELICA; BACIVAROF, I.C., *Modelarea fiabilității software-ului. O analiză critică*, Automatică, Management, Calculatoare, 36, Editura Tehnică, București, 1983, p. 116—130.

CAPITOLUL 5

EVALUAREA PERFORMANȚELOR DE FIABILITATE PENTRU SISTEMELE TOLERANTE LA DEFECTĂRI

5.1. INTRODUCERE

În general, evaluarea performanțelor de fiabilitate pentru un sistem constă în identificarea expresiei analitice sau a valorilor numerice ale indicatorilor de fiabilitate ai sistemului — de obicei, funcția de fiabilitate, $R(t)$, sau funcția de repartiție a timpului de funcționare fără defectări, $F(t)$, atunci cînd se cunosc indicatorii de fiabilitate ai elementelor sale componente.

În scopul evidențierii performanțelor de fiabilitate ale sistemelor tolerante la defectări pot fi folosiți atît indicatori generali de fiabilitate, cît și indicatori de fiabilitate specifici acestei categorii de sisteme. Astfel, în analizele prezentate în continuare pentru sistemele hardware cu structură tolerantă la defectări se vor utiliza următorii indicatori de fiabilitate:

— funcția de fiabilitate, $R(t)$;
— funcția de repartiție a timpului de funcționare, $F(t)$;
— media timpului de funcționare pînă la prima defectare m, (MTFF);
— media timpului de funcționare între defectări, MTBF;
— disponibilitatea sistemului, $A(t)$, respectiv coeficientul de disponibilitate, A;
— securitatea sistemului, $\mathcal{S}(t)$;
— credibilitatea sistemului, $\mathcal{C}(t)$;
— factorul de acoperire a defectărilor ș.a.

Pentru caracterizarea performanțelor de fiabilitate ale sistemelor software tolerante la defectări pot fi utilizați — între alții — ca indicatori de fiabilitate:

— funcția de fiabilitate a software-ului, $R_s(t)$;
— rata erorilor de software, $z_s(t)$;
— timpul mediu între erorile de software, TMIE;
— disponibilitatea sistemului software, $A_s(t)$.

Construcția modelelor matematice pentru evaluarea indicatorilor de fiabilitate ai sistemelor se bazează pe noțiunea de *defectare*, înțeleasă ca ieșirea a cel puțin uneia dintre performanțele sistemului din limitele spe-

cificate. Această definire convențională a defectării presupune o perfectă cunoaștere a performanțelor sistemului și a domeniilor în care trebuie să fie situate acestea în vederea îndeplinirii cu succes a unei anumite misiuni. Însă, nu întotdeauna se evidențiază imediat care performanțe ale unui sistem sînt relevante față de o anumită aplicație, iar luarea în considerare a totalității performanțelor unui sistem este importantă chiar în cazul sistemelor relativ simple.

De exemplu, pentru sistemele de prelucrare a informației sensul noțiunii de funcționare corectă diferă de cel atribuit altor sisteme electronice, pentru că funcționarea corectă înseamnă execuție corectă a programului, performanțe corecte ale funcțiilor intrare/ieșire și protecția datelor de bază la erori. Așadar, cerințe mult mai severe decît acelea privind funcționarea componentelor unui radioreceptor, de pildă. Zgomotul electric al unui tranzistor este acceptat într-un amplificator din compunerea unui radioreceptor. Dacă tranzistorul respectiv urmează să controleze intrarea unui bistabil al unui registru de instrucțiuni dintr-un procesor, chiar dacă zgomotul va altera numai un bit al cuvîntului instrucțiune, este foarte probabil că întreg programul va fi compromis. Faptul că tranzistorul nu se defectează, iar zgomotul durează numai un interval mic de timp nu poate fi neglijat dacă se are în vedere realizarea unui sistem fiabil de prelucrare a informației.

Aceste diferențe nu sînt avute în vedere de modelele matematice din teoria generală a fiabilității [15], [42], [43], motiv pentru care a devenit necesară elaborarea unor modele de analiză a fiabilității *specifice* sistemelor digitale de mare răspundere funcțională și implicit sistemelor tolerante la defectări.

O proiectare bazată pe evaluarea performanțelor de fiabilitate face posibilă motivarea alegerii anumitor tehnici specifice de implementare a unui sistem. Astfel, în acest context sînt analizate tehnicile pentru tolerarea defectărilor și realizarea validării funcționării în corelație cu progresele tehnologice, ca și cu evaluarea cerințelor utilizatorilor. De o mare importanță pentru utilizatori, alături de indicatorii de fiabilitate (disponibilitate, securitate) este evaluarea *încrederii în sistem*, care reunește indicatorii anteriori în conceptul *dependability* [32].

Pentru modelarea fiabilității sistemelor complexe și evaluarea acestor indicatori pot fi utilizate metodologii bazate pe procesele Markov, ceea ce permite atît un studiu unitar fundamentat pe indicatorii menționați mai sus, cît și analiza fiabilității sistemelor a căror rată de defectare (respectiv reparare) nu este constantă în timp. Astfel de abordări au fost integrate în pachetul de programe SURF [28] [33], care au fost utilizate pentru evaluarea performanțelor atît ale unor sisteme de calcul de înaltă securitate [31], cît și ale unor supercalculatoare [1].

În cazul dependențelor stocastice problemele de evaluare nu mai pot fi rezolvate cu metoda lanțurilor Markov, ci utilizîndu-se abordări bazate pe rețelele stocastice Petri [16].

În ceea ce privește evaluarea performanțelor sistemelor software cercetările au fost concentrate în primul rînd asupra estimării disponibilității acestora [23], [29], [4], luîndu-se în considerație atît erorile software-ului, cît și defectările suportului material.

În general, cercetările din domeniul evaluării fiabilității sistemelor sînt axate pe două direcții de bază: o abordare *analitică*, fundamentată pe dezvoltarea unui model matematic, și o abordare *experimentală*, prin care defectele sînt înserate în modele de simulare ale sistemului, iar caracteristicile de fiabilitate sînt estimate pe baza datelor statistice. De exemplu, proiectul EVE de estimare a indicatorilor de fiabilitate pentru unele sisteme tolerante la defectări [2], [26] utilizează pentru validarea rezultatelor — în afara unei evaluări bazate pe modele Markov — și o metodă de injecție a defectărilor în sistemul simulat, ceea ce face posibilă o abordare mai realistă.

În acest capitol sînt tratate cîteva modele analitice de evaluare a performanțelor de fiabilitate, adecvate sistemelor hardware cu structură redondantă, forma principală de implementare a toleranței la defectări.

5.2. INDICATORI DE FIABILITATE

5.2.1. CONSIDERAȚII GENERALE

Performanțele de fiabilitate ale unui sistem pot fi descrise matematic la nivel *global*, cînd nu se ține seama de structura internă a sistemului, sau la nivel *structural*, atunci cînd se iau în considerare elementele sistemului și relațiile între ele. În ambele cazuri caracterizarea performanțelor de fiabilitate se va face utilizîndu-se teoria probabilităților. Descrierea acestor performanțe de fiabilitate poate fi efectuată într-un spațiu unidimensional prin introducerea noțiunii de defectare, înțeleasă ca depășire a limitelor specificate de către cel puțin una din performanțele sistemului.

Pentru elaborarea unor modele matematice care să conducă la rezultate credibile asupra acestor performanțe este important ca ipotezele care stau la baza acestor modele să aproximeze cît mai bine sistemul fizic.

În cele ce urmează fenomenul *defectare* se consideră ca avînd la origine trei clase de evenimente fizice care afectează hardware-ul sau software-ul sistemului:

— defectări permanente ale componentelor hardware;

— defectări funcționale temporare (cvasitemporare) ale sistemului;

— erori de concepție.

În analizele următoare se vor avea în vedere aceste trei clase de evenimente. Modelarea fiabilității pentru partea hardware va fi limitată la partea electrică a sistemului analizat. Se va lua în considerare cazul în care numărul de stări ale sistemului va fi relativ mic, fiecare stare avînd o reprezentare distinctă.

Indicatorii de fiabilitate aplicați diferitelor arhitecturi de sisteme tolerante la defectări sînt în general cei definiți în Anexa A.1; aceste definiții sînt adecvate pentru sistemele tolerante la defectări cu structură redondantă statică sau dinamică, dar nu se consideră potrivite pentru evaluarea caracteristicilor de fiabilitate ale sistemelor autotestabile (cu semnal de stop) sau ale sistemelor reconfigurabile de tip *gracefuly degradation* [35].

Atunci cînd se iau în considerare restabilirea imperfectă ca și detecția imperfectă a defectărilor, acestea se pot modela prin conceptul de acope-

rire (*coverage*) [5] [40], definit ca o proporție a unui defect de la care un sistem se restabilește automat. Pentru cele mai multe categorii de sisteme reconfigurabile este foarte important ca acestea să-și reconfigureze automat structura dacă în oricare unitate a lor este detectat un defect. În general, sistemele tolerante la defectări îndeplinesc misiuni extrem de importante care nu permit întreruperi. Pentru o evaluare mai realistă a performanțelor de fiabilitate se impune dezvoltarea unor modele care să țină seama de factorul de acoperire menționat, aceasta fiind totodată o soluție pentru modelarea defectărilor tranzitorii sau intermitente.

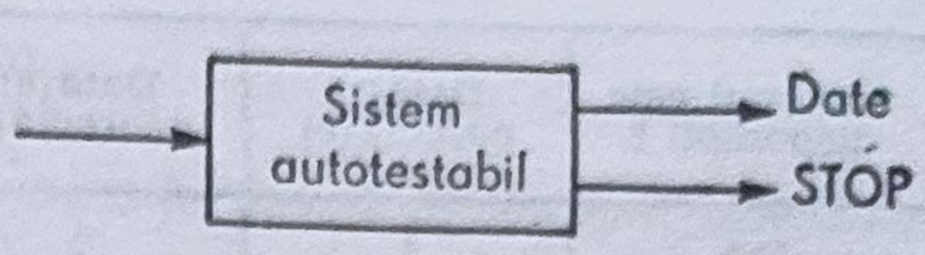


Fig. 5.1. Reprezentarea schematică a unui sistem autotestabil.

Sistemele tolerante la defectări și autotestabile care asigură o înaltă securitate în funcționare comportă două tipuri de ieșiri — fig. 5.1 — vezi și capitolul 3: o ieșire de date și, respectiv, una indicatoare de defect, care poate declanșa oprirea sistemului. În § 5.2.2 se vor defini o serie de indicatori caracteristici pentru astfel de sisteme.

5.2.2. INDICATORI DE FIABILITATE PENTRU SISTEMELE AUTOTESTABILE

Definiția 5.1. *Securitatea* sistemului reprezintă probabilitatea ca la ieșirea acestuia să nu apară o informație eronată nedetectată, adică:

$$\mathcal{S} = 1 - \Pr \text{ (informația dată de sistem este eronată } \textit{și} \text{ semnalul de STOP nu a apărut).} \tag{5.1.}$$

Definiția 5.2. *Fiabilitatea* sistemului reprezintă probabilitatea de a executa corect misiunea sa pe durata afectată acesteia, adică:

$$R = \Pr \text{ (informația dată de sistem este corectă } \textit{și} \text{ semnalul de STOP nu a apărut).} \tag{5.2}$$

Definiția 5.3. *Disponibilitatea* sistemului reprezintă probabilitatea ca acesta să fie disponibil la un moment de timp dat, adică probabilitatea ca semnalul de STOP să nu fi apărut:

$$A = \Pr \text{ (semnalul STOP nu a apărut).} \tag{5.3}$$

Definiția 5.4. *Credibilitatea* sistemului reprezintă probabilitatea ca informația dată să fie corectă, condiționată de faptul că sistemul este disponibil, adică:

$$\mathcal{C}(t) = \Pr \text{ (informația este corectă/sistemul este disponibil).} \tag{5.4}$$

Credibilitatea definită în acest fel, este o măsură a *încrederii* acordate informației disponibile la ieșirea sistemului.

Definiția dată aici pentru disponibilitate nu implică faptul că semnalele de ieșire funcționale sînt corecte, ci numai cerința ca semnalul de STOP să nu apară. Semnalele de ieșire vor fi corecte dacă securitatea sistemului este perfectă. În aceasta constă diferența dintre definiția dată disponibilității pentru sistemele reparabile și definiția dată aici pentru

Sistemul este disponibil ?	Dată (e) corectă (e)	Dată (e) incorectă (e)
DA	1	2
?	3	4
NU	5	6

Fig. 5.2. Justificarea definiţiilor pentru indicatorii specifici sistemelor autotestabile.

sisteme nereparabile; în cazul definiţiei „clasice" se subînţelegea că sistemul funcţionează corect dacă este disponibil.

Cei patru indicatori prezentaţi mai sus sînt suficienţi pentru caracterizarea unui sistem autotestabil. Într-adevăr, considerîndu-se o schemă a stărilor posibile ale unui sistem autotestabil, cum este cea prezentată în figura 5.2, relaţiile de definiţie 5.1 ÷ 5.4 se pot explicita cu ajutorul următoarelor ecuaţii:

$$\left.\begin{aligned} \mathcal{S} &= \Pr(1 \cup 3 \cup 5 \cup 6), \\ R &= \Pr(1), \\ A &= \Pr(1 \cup 2); \\ \mathcal{C} &= \Pr(1 \cup 2 \cup 3 \cup 4 \cup 5 \mid 1), \end{aligned}\right\} \tag{5.5}$$

iar

$$\Pr(1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6) = 1.$$

S-au notat cu 1,..., 6 cele şase stări posibile ale sistemului.

Particularizîndu-se cele patru definiţii prezentate aici în cazul sistemelor care nu au semnal de STOP (cu structura redondantă sau nu) rezultă că disponibilitatea este egală cu 1, iar securitatea, fiabilitatea misiunii şi credibilitatea sînt egale între ele şi egale cu fiabilitatea sistemului în sensul definiţiei date în Anexa 1.

5.3. ANALIZA PERFORMANŢELOR DE FIABILITATE ALE SISTEMELOR TOLERANTE LA DEFECTĂRI ÎN IPOTEZA DEFECTĂRILOR DE TIP PERMANENT

5.3.1. CONSIDERAŢII ASUPRA PROBLEMELOR ANALIZEI FIABILITĂŢII SISTEMELOR TOLERANTE LA DEFECTĂRI

Sistemele intolerante la defectări sînt sisteme cu model logic de fiabilitate de tip serie, la care fiecare componentă trebuie să funcţioneze corespunzător pentru a asigura funcţionarea sistemului la performanţele cerute. Nivelul înalt de fiabilitate pentru astfel de sisteme este atins atunci cînd se utilizează componente foarte fiabile şi/sau tehnici speciale de interconexiune şi asamblare a elementelor componente. Pentru realizarea unor astfel de sisteme ultrafiabile apar constrîngeri de cost şi performanţe, care — aşa cum s-a prezentat în capitolul 1 — pot fi rezolvate utilizîndu-se sistemele cu structură redondantă.

Apare necesitatea practică a comparării performanţelor de fiabilitate ale sistemelor cu structură redondantă în raport cu cele ale sistemelor neredondante.

Indicatorii de fiabilitate — funcția de fiabilitate, $R(t)$, și media timpului de funcționare, m —, utili la compararea sistemelor neredondante între ele, nu conduc întotdeauna la rezultate clare pentru proiectant în această situație.

Pentru exemplificare, în figura 5.3 se dau graficele funcțiilor de fiabilitate ale unui sistem neredondant (graficul *a*) și ale unui sistem cu structură redondantă logică majoritară (graficul *b*). Din analiza acestor reprezentări grafice se observă că media timpului de bună funcționare — măsurată ca arie a suprafeței cuprinse între graficul lui $R(t)$ și axa Ot— nu poate fi utilizată ca indicator de comparație a performanțelor de fiabilitate ale celor două sisteme (redondant și neredondant). Problema poate fi rezolvată [5] [7] folosindu-se pentru comparație probabilitățile de defectare, măsurate prin valorile funcțiilor de repartiție, sau cuantilele timpilor de bună funcționare. În capitolul 2 s-au introdus indicatori specifici sistemelor redondante pentru compararea acestora.

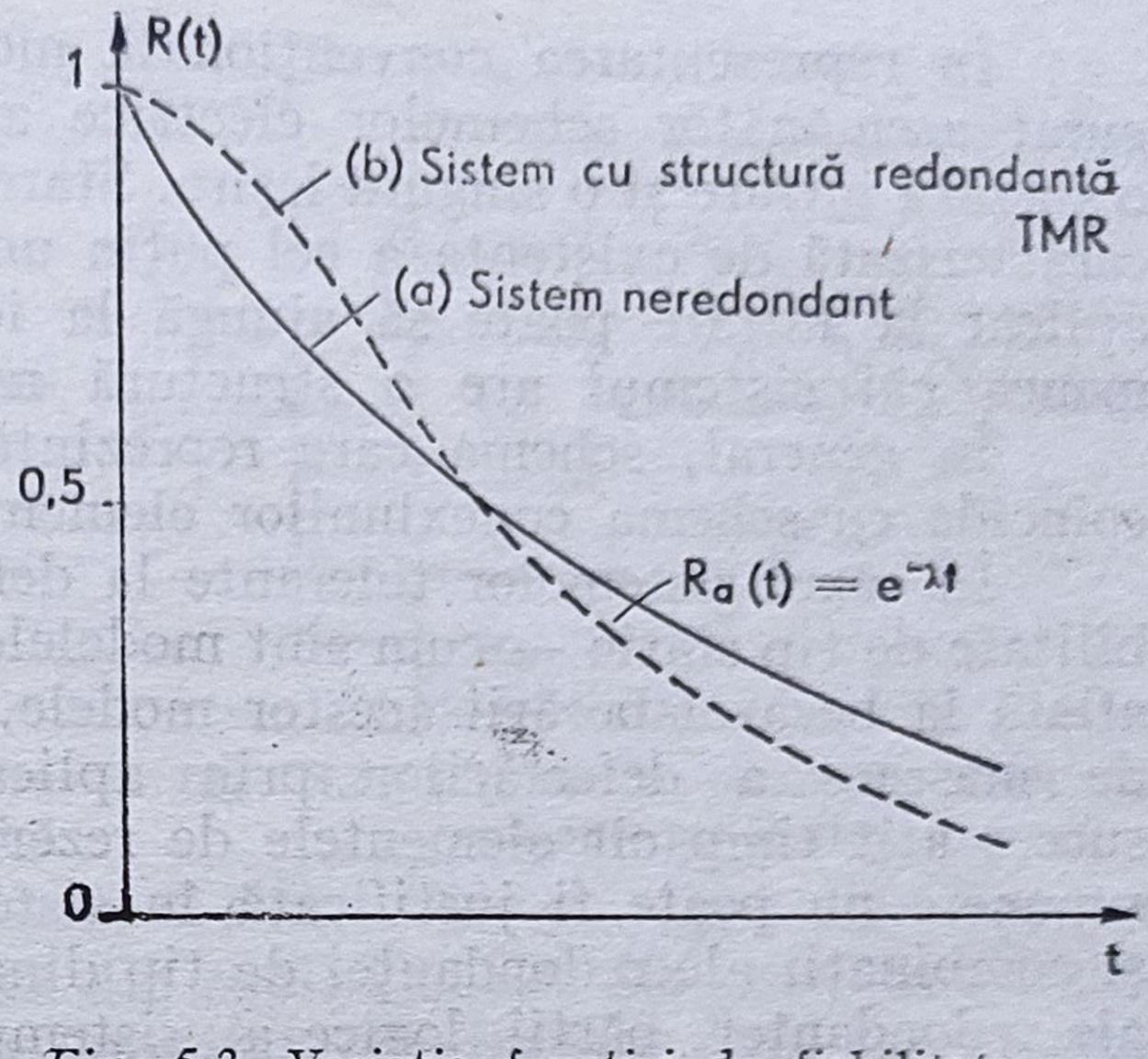


Fig. 5.3. Variația funcției de fiabilitate cu timpul misiunii:
a — sistem neredondant; *b* — sistem cu structură redondantă logică majoritară 2 din 3.

Una dintre direcțiile principale de cercetare în domeniul evaluării performanțelor de fiabilitate ale sistemelor tolerante la defectări constă în analiza și evaluarea fiabilității structurilor redondante.

Analiza fiabilității unui sistem [22] presupune mai multe etape care se interacționează reciproc, și anume:

— determinarea, pornind de la modelul de funcționare real al sistemului, a modelului său de fiabilitate care evidențiază condițiile ce conduc la buna funcționare a acestuia;

— stabilirea metodologiei de analiză a fiabilității adecvată modelului adoptat;

— evaluarea propriu-zisă a funcției de fiabilitate sau a altor indicatori de fiabilitate, ținîndu-se seama de datele privind fiabilitatea elementelor sale componente și de modelul de fiabilitate adoptat.

Majoritatea lucrărilor de specialitate referitoare la evaluarea performanțelor de fiabilitate ale structurilor redondante au în vedere modele care pornesc de la faptul că mascarea defectelor are loc instantaneu, iar defectele copiilor individuale ale modulelor funcționale sînt independente.

Modelul de fiabilitate adoptat trebuie să redea în mod univoc relația dintre fiabilitatea sistemului și cea a elementelor sale componente. În acest scop pot fi utilizate reprezentări bazate pe succes — bună funcționare — sau insucces — defectare. În majoritatea analizelor de fiabilitate sînt folosite modele logice (structurale) de fiabilitate, care constituie reprezentări bazate pe succes ale legăturilor de fiabilitate între elementele sistemului, ținîndu-se seama de gradul în care acestea influențează starea de funcționare a sistemului analizat.

În reprezentarea convențională modelul logic de fiabilitate este figurat asemănător schemelor electrice avînd intrări și ieșiri — de obicei o singură intrare și o singură ieșire. Starea de funcționare a sistemului este caracterizată de existența a cel puțin unei căi prin care semnalul ipotetic aplicat la intrare poate să ajungă la ieșire. Dacă există mai multe asemenea căi sistemul are o structură redondantă.

În general, schema care reprezintă modelul logic de fiabilitate nu coincide cu schema conexiunilor elementelor sale.

În cazul sistemelor tolerante la defectări aplicarea modelelor de fiabilitate de tip static — cum sînt modelele logice — este limitată de ipoteza aflată la baza elaborării acestor modele, în conformitate cu care acțiunea de mascare a defectărilor prin aplicarea redondanței este realizată cu succes atît timp cît elementele de rezervă nu sînt epuizate. Această presupunere nu poate fi justificată în sistemele care folosesc diferite forme și combinații ale redondanței de tip dinamic la nivel hardware sau forme ale redondanței părții logice a sistemului și ale redondanței de timp. Această limitare este soluționată prin utilizarea unor modele de evaluare a fiabilității de tip dinamic, cum ar fi modelul lanțurilor Markov.

Modelele markoviene permit evidențierea evoluției în timp a unui sistem din punctul de vedere al fiabilității. Cele bazate pe procesele Markov sînt generale, pornindu-se de la ele fiind posibilă descrierea sistemelor atît la nivel global, cît și la nivelul structurilor interne.

De menționat că în abordările practice trebuie avut în vedere faptul că analiza fiabilității bazată pe modelarea Markov este suficient de laborioasă și de aceea pentru unele cazuri care ar putea fi rezolvate mult mai simplu folosindu-se alte metode (de exemplu, plecîndu-se direct de la modelul logic la fiabilitate) ea nu trebuie utilizată în mod abuziv.

Rezultă că modelele de fiabilitate statice sînt utile în cazul sistemelor cu redondanță protectivă de tip static, cînd modelele redondante sînt folosite pentru mascarea efectului defectelor materiale, ieșirile sistemului rămînînd neafectate atît timp cît protecția este efectivă.

Utilizarea modelelor statice pentru cazul structurilor de tip dinamic este echivalent cu a considera o probabilitate maximă a succesului acțiunilor de detecție a defectărilor și refacerii sistemului după orice defectare. Din acest motiv, folosirea acestor modele pentru o structură redondantă de tip dinamic poate conduce la previziuni optimiste de fiabilitate. În realitate o detecție imperfectă a defectărilor sau o reconfigurare imperfectă pot lăsa, utilizîndu-se această structură redondantă, cîteva rezerve nefolosite, astfel încît nu se mai ajunge la o fiabilitate foarte înaltă. Efectul acestor imperfecțiuni poate fi modelat prin introducerea conceptului de probabilitate a reconfigurării cu succes a sistemului atunci cînd a avut loc un defect [17].

5.3.2. FUNCȚIA DE FIABILITATE PENTRU STRUCTURA REDONDANTĂ LOGICĂ MAJORITARĂ

Funcția de fiabilitate a unei structuri redondante logică majoritară de tip k din n poate fi modelată pornindu-se de la modelul logic de fiabilitate (fig. 5.4) după cum urmează:

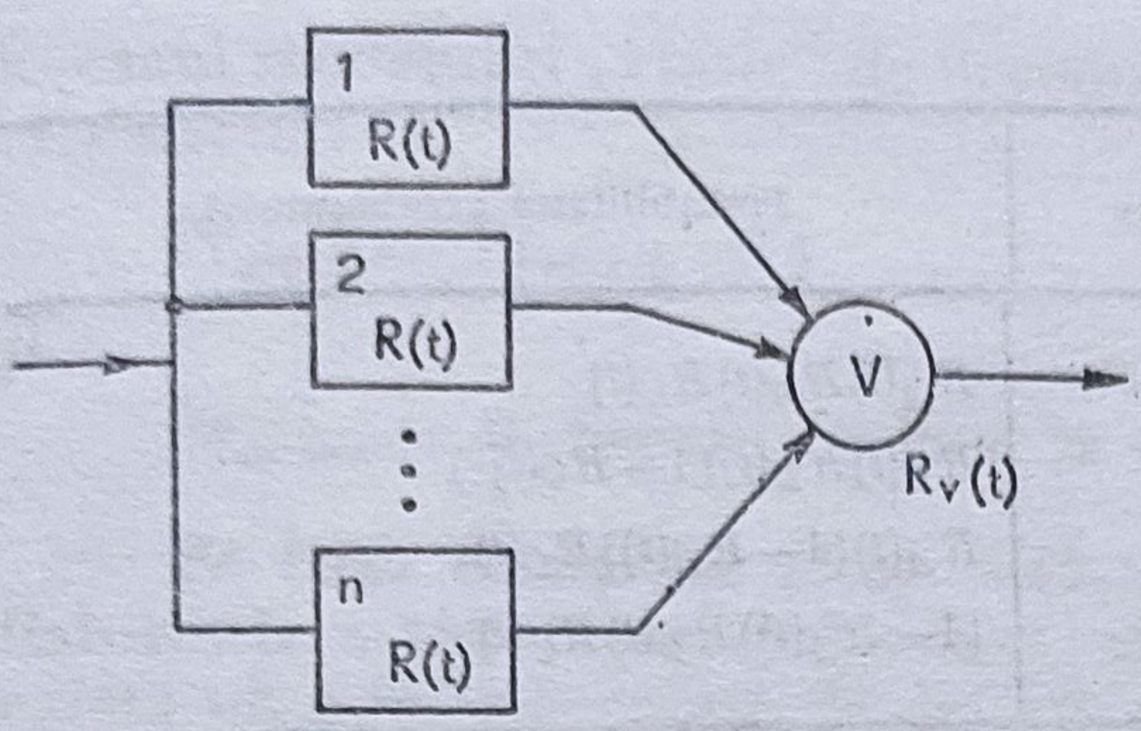


Fig. 5.4. Modelul logic de fiabilitate pentru sistemul cu structură redondantă logică majoritară k din n.

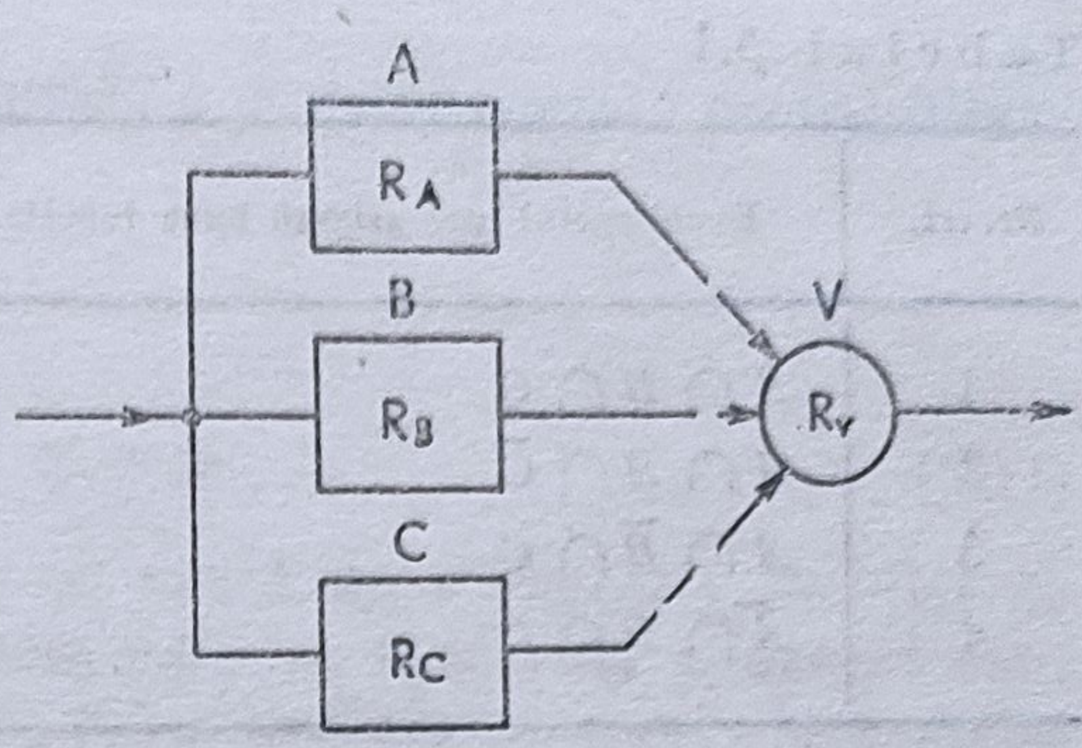


Fig. 5.5. Sistem de structură TMR cu elemente neidentice.

Se consideră structura formată din copii identice ale sistemului funcțional de bază, avînd funcția de fiabilitate de valoare $R(t)$, iar voterul cu funcția de fiabilitate $R_V(t)$. Pentru ca sistemul să fie în bună stare de funcționare, trebuie să funcționeze corect cel puțin k unități funcționale și voterul V.

Probabilitatea $R_{NMR}(t)$ corespunzător căreia cel puțin k unități funcționale din cele n, precum și voterul să fie în stare de funcționare (restul unităților putînd fi defecte) este:

$$R_{NMR}(t) = R_V(t)\left[\sum_{i=k}^{n} C_n^i R(t)^i (1 - R(t))^{n-i}\right] \tag{5.6}$$

Relația poate fi particularizată pentru cazul celei mai uzuale structuri redondante de tip logică majoritară, structura logică majoritară 2 din 3 (TMR), rezultînd:

$$R_{TMR}(t) = R_V(t)(3R(t)^2 - 2R(t)^3). \tag{5.7}$$

Ecuațiile (5.6) și (5.7) sînt aplicabile atunci cînd modulele funcționale sînt identice. Dacă însă modulele 1,..., n, deși îndeplinesc aceleași funcții, au scheme și — evident — fiabilități diferite (de exemplu, pentru a se evita erorile de proiectare în sistemele software sau chiar în cele hardware), calculul funcției de fiabilitate devine laborios. În asemenea situații este necesar să se apeleze la alte metode pentru evaluarea fiabilității acestui tip de structură redondantă, cum ar fi de exemplu metoda enumerării exhaustive a stărilor sistemului [11].

E x e m p l u l 5.1 Se consideră o structură redondantă de tip 2 din 3, formată din unitățile funcționale A, B, C [20] cu funcțiile de fiabilitate $R_A(t)$, $R_B(t)$, $R_C(t)$ (fig. 5.5). În tabelul 5. 1 sînt indicate posibilitățile de funcționare corectă ale sistemului și probabilitățile de realizare a acestor evenimente. Reuniunea celor patru evenimente asigură buna funcționare a sistemului. Evenimentele sînt incompatibile între ele, astfel că probabilitatea de bună funcționare a sistemului redondant este:

$$R_{TMR}(t) = R_V(t)[R_A(t)R_B(t)R_C(t) + R_A(t)R_B(t)(1 - R_C(t)) + R_A(t)R_C(t)(1 - R_B(t)) + R_B(t)R_C(t)(1 - R_A(t))] = R_V(t)[(R_A(t)R_B(t) + R_A(t)R_C(t) + R_B(t)R_C(t) - 2R_A(t)R_B(t)R_C(t)].$$

Tabelul 5.1

Nr. crt.	Evenimentul care asigură buna funcționare	Probabilitatea evenimentului
1	$A \cap B \cap C$	$R_A(t)R_B(t)R_C(t)$
2	$A \cap B \cap \bar{C}$	$R_A(t)R_B(t)(1-R_C(t))$
3	$A \cap \bar{B} \cap C$	$R_A(t)(1-R_B(t))R_C(t)$
4	$\bar{A} \cap B \cap C$	$(1-R_A(t))R_B(t)R_C(t)$

Atunci cînd unitățile funcționale sînt circuite logice expresiile date funcției de fiabilitate de ecuațiile (5.6) și (5.7) conduc la rezultate pesimiste. Această situație apare deoarece în asemenea sisteme există situații de compensare a erorilor, pe care modelarea funcției de fiabilitate utilizîndu-se ecuațiile de mai sus nu le ia în considerare. De exemplu, așa cum s-a evidențiat în §2.5, pentru structura redondantă logică majoritară de tip 2 din 3 blocarea în „1" a unui modul funcțional și blocarea în „0" a altui modul funcțional nu vor conduce la defectarea schemei, deși sînt defecte două module funcționale. De aceea, în scopul evaluării și comparării unor structuri redondante de acest tip, în cadrul aceluiași paragraf a fost dezvoltat — pe baza metodei tăieturilor minimale — un model denumit „modelul modurilor de defectare", care ține seama de aceste compensări.

În același sens, pornindu-se de la [17], se poate dezvolta un model de evaluare a funcției de fiabilitate pentru structura redondantă logică majoritară de tip 2 din 3, cu luarea în considerare a posibilității compensării unui număr de defecte în fiecare modul.

Astfel, se poate scrie probabilitatea evenimentului de a avea m defecte într-un modul, $n < m$ defecte în cel de-al doilea și $r < n$ defecte în cel de-al treilea modul și cele trei module să funcționeze:

$$\Pr = R(t)\frac{(\lambda t)^m}{m!} \cdot R(t)\frac{(\lambda t)^n}{n!} \cdot R(t)\frac{(\lambda t)^r}{r!}, \tag{5.8}$$

unde λ este rata de defectare a unui modul.

Pentru oricare triplet m, n, r există un număr de permutări, P_{mnr}, pentru atribuirea numerelor m, n, r celor trei module:

$$P_{mnr} = \begin{cases} 1, & m = n = r; \\ 3, & m = n \text{ sau } n = r; \\ 6, & m > n > r. \end{cases} \tag{5.9}$$

Fiecărui triplet i se asociază o probabilitate condiționată $\Pr_{m,n,r}$ definită astfel:

$\Pr_{m,n,r} = \Pr$ (sistemul funcționează corect dacă sînt m, n respectiv r defecte); (5.10)

în cazul modelului „clasic" de necompensare a erorilor această probabilitate ia valorile:

$$Pr_{0,0,0} = Pr_{m,0,0} = 1$$

și

$$Pr_{m,n,r} = 0 \text{ pentru } m \geqslant n \geqslant r, \; n > 0. \tag{5.11}$$

Pe baza ecuațiilor (5.8) ÷ (5.10) se poate scrie expresia funcției de fiabilitate a sistemului redondant, sistem care va funcționa și atunci cînd există un număr oarecare de defecte în primul modul, un număr $n < m$ defecte în cel de-al doilea și un număr $r \leqslant n$ în cel de-al treilea modul:

$$R_{TMR}(t) = \sum_{m=0}^{\infty} \sum_{n=0}^{m} \sum_{r=0}^{n} P_{mnr} \, Pr_{m,n,r} \, R(t) \frac{(\lambda t)^m}{m!} \; R(t) \frac{(\lambda t)^n}{n!} R(t) \frac{(\lambda t)^r}{r!} =$$

$$= R(t)^3 \sum_{m=0}^{\infty} \sum_{n=0}^{m} \sum_{r=0}^{n} P_{mnr} \, Pr_{m,n,r} \frac{(\lambda t)^{m+n+r}}{m!n!r!} \tag{5.12}$$

Explicitînd suma se obține:

$$R_{TMR}(t) = P_{000} Pr_{0,0,0} \, R(t)^3 + R(t)^3 \sum_{m=1}^{\infty} Pm_{00} \, Pr_{m,0,0} \frac{(\lambda t)^m}{m!} +$$

$$+ \sum_{m=1}^{\infty} \sum_{n=1}^{m} \sum_{r=0}^{n} P_{mnr} \, Pr_{m,n,r} \frac{(\lambda t)^{m+n+r}}{m!n!r!} \tag{5.13}$$

Deoarece în cazul sistemelor electronice sau al celor complexe se poate considera o repartiție exponențială a timpului de funcționare, caz în care $R(t) = e^{-\lambda t}$, rezultă:

$$\frac{1}{R(t)} = \sum_{m=0}^{\infty} \frac{(\lambda t)^m}{m!} = 1 + \sum_{m=1}^{\infty} \frac{(\lambda t)^m}{m!}$$

sau

$$\sum_{m=1}^{\infty} \frac{(\lambda t)^m}{m!} = \frac{1 - R(t)}{R(t)} \tag{5.14}$$

Ținîndu-se seama de (5.11) și (5.14), ecuația (5.13) devine:

$$R_{TMR}(t) = R(t)^3 + 3R(t)^2(1 - R(t)) +$$

$$\oplus \; R(t)^3 \sum_{m=1}^{\infty} \sum_{n=1}^{m} \sum_{r=0}^{n} P_{mnr} \; Pr_{m,n,r} \frac{(\lambda t)^{m+n+r}}{m!n!r!} \tag{5.15}$$

Pentru evaluarea funcției de fiabilitate rămîne de rezolvat problema determinării probabilității $Pr_{m,n,r}$ pentru cazuri concrete.

De menționat că această modelare dată funcției de fiabilitate nu apelează la structura concretă a sistemului decît pentru determinarea probabilității $Pr_{m,n,r}$ cum se întîmpla în cazul modelului modurilor de defectare.

5.3.3. FUNCȚIA DE FIABILITATE PENTRU UN SISTEM CU STRUCTURĂ REDONDANTĂ DE TIP LOGICĂ CUADRUPLĂ

Modul de implementare a acestui tip de redondanță a fost prezentat detaliat în § 2.2.3. S-a arătat că schema constă în interconectarea de porți logice, astfel încît grupul celor patru ieșiri ale sistemului să mascheze o serie de defecte din structura acestuia. Pentru ieșire, sistemul va apărea în bună stare de funcționare dacă nu se defectează mai mult de unul dintre elementele nivelurilor I, respectiv III și nici un element din nivelul II (a se vedea figura 2.12). Rezultă de aici expresia funcției de fiabilitate a unui astfel de sistem:

$$R_{SR}(t) = R_{II}(t)^4\left[\sum_{i=0}^{1} C_4^i\ R_I(t)^{4-i}\ (1-R_I(t))^i\right]\times$$

$$\times\left[\sum_{j=0}^{1} C_4^j\ R_{III}\ (t)^{4-j}\,(1-R_{III}\ (t))^j\right], \qquad (5.16)$$

unde $R_I(t)$, $R_{II}(t)$ și $R_{III}(t)$ desemnează funcțiile de fiabilitate ale elementelor nivelelor I, II și, respectiv, III.

5.3.4. FUNCȚIA DE FIABILITATE PENTRU UN SISTEM CU STRUCTURĂ REDONDANTĂ DE COMUTAȚIE

Modelul logic de fiabilitate al acestei structuri este indicat în figura 5.6. S-a considerat cazul uzual cînd există un modul funcțional și unul în rezervă. Se notează cu $R_1(t)$ funcția de fiabilitate a modulului funcțional de bază, cu $R_2(t)$ funcția de fiabilitate a modulului de rezervă atunci cînd acesta este în funcțiune și cu $R_{2r}(t)$ funcția de fiabilitate a modulului de rezervă atunci cînd el este în așteptare. Modulul de rezervă poate — — sau nu — să fie identic cu modulul funcțional.

Buna funcționare a sistemului este asigurată de realizarea următoarelor evenimente:

(1) Sistemul de bază funcționează bine pe durata de timp 0, t; probabilitatea acestui eveniment este:

$$\mathrm{Pr}_1 = R_1(t);$$

(2) Sistemul de bază se defectează la un moment de timp anterior momentului t, fie acesta momentul τ; sistemul în rezervă este în stare de bună funcționare și funcționează bine de la momentul τ pînă la momentul t.

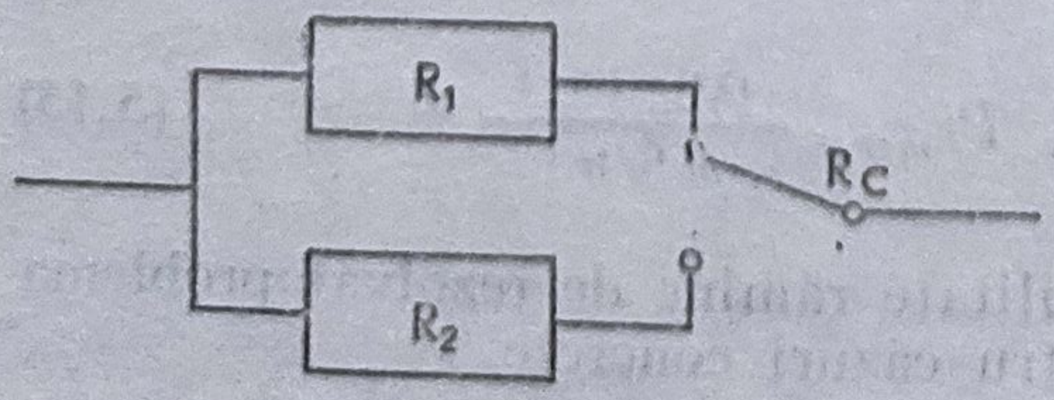


Fig. 5.6. Modelul logic de fiabilitate pentru sistemul cu structură redondantă de comutație.

Probabilitatea defectării sistemului de bază în intervalul de timp infinit mic τ, $\tau + d\tau$ este $f(\tau)\,d\tau$, iar probabilitatea defectării sistemului de bază la momentul τ și a funcționării sistemului de

rezervă de la momentul τ la momentul t, sistemul de rezervă fiind în stare de funcțiune la momentul τ, este:

$$f(\tau)R_{2n}(\tau)R_2(t-\tau)d\tau.$$

Momentul τ poate fi oricare pe intervalul de timp 0, t, astfel încît realizarea evenimentului compus (2) este:

$$Pr_2 = \int_0^t f(\tau)\, R_{2r}(\tau)R_2(t-\tau)d\tau.$$

Evenimentele (1) și (2) sînt incompatibile între ele, așa încît probabilitatea bunei funcționări a acestei structuri redondante se poate scrie ca fiind:

$$R_{SR}(t) = R_1(t) + \int_0^t f_i(\tau)R_{2r}(\tau)\, R_2(t-\tau)d\tau. \qquad (5.17)$$

Astfel modelată funcția de fiabilitate, fără a se lua în considerare posibilitatea — practică — a unei reconfigurări imperfecte, conduce la evidențierea unor performanțe optimiste ale structurii analizate.

Utilizîndu-se conceptul de reconfigurare (cu succes) a structurii sistemului și probabilitatea aferentă, c, se poate da o modelare mai realistă a funcției de fiabilitate corespunzătoare acestei structuri.

În afară de timpul misiunii, t, rata de defectare, λ, și rata de defectare în stocare, λ_r, parametrii ce caracterizează buna funcționare a sistemului cu această structură în prezența defectărilor sînt: numărul de module funcționale aflate în rezervă, r, numărul minim de module funcționale care trebuie să funcționeze pentru ca sistemul să fie în stare de funcționare, k, numărul de defectări tolerate de sistem, l. Prin urmare funcția de fiabilitate va fi dependentă de toți acești parametri:

$$R_{SR} = \varphi(r,l,k,\lambda,\lambda_r,c). \qquad (5.18)$$

c avînd semnificația probabilității detectării cu succes a unei defectări a sistemului și comutării unei rezerve.

Considerîndu-se repartiția timpului de funcționare exponențială și urmînd un raționament analog celui dezvoltat anterior, se poate detalia funcția de fiabilitate (5.18). Se obține următoarea relație de recurență:

$$R_{SR}(r,l,k,\lambda,\lambda_r,t,c) = R_{SR}(r-1,l,k,\lambda,\lambda_r,t,c) +$$

$$+\int_0^t c^r \frac{\mathrm{d}}{\mathrm{d}t}\Big[1 - R_{SR}(r-1,l,k,\lambda,\lambda_r,t,c)e^{-\lambda r\tau}\, e^{-\lambda(t-\tau)}\, d\tau, \qquad (5.19)$$

unde c^r exprimă probabilitatea a r reconfigurări efectuate cu succes în sistem, $e^{-\lambda r\tau}$ — probabilitatea că modulul r de rezervă se află în stare de funcționare în momentul cînd este introdus în sistem; iar $e^{-\lambda(t-\tau)}$ — probabilitatea că modulul r va realiza misiunea din momentul conectării sale în sistem pînă la momentul t.

5.3.5. FUNCȚIA DE FIABILITATE PENTRU UN SISTEM CU STRUCTURĂ REDONDANTĂ PRIN CODARE

În § 2.2.5 a fost prezentată o astfel de structură rezultată prin mărirea setului de stări ale unui automat de la 2^k la 2^n — prin creșterea numărului de circuite aferente — , astfel încît fiecăreia din cele 2^k stări îi va reveni un număr de 2^{n-k} stări redondante din cele 2^n . Dacă se defectează un număr de elemente, iar cele 2^{n-k} stări nu vor fi afectate în totalitate, vor rezulta funcții răspuns ale sistemului în prezența uneia sau a mai multor funcții de stare eronate.

Se consideră schema de implementare a redondanței indicată în figura 2.19. În scopul facilitării analizei de fiabilitate care urmează, fiecare grup M_i, C_i este înlocuit de o unitate echivalentă, H_i, rezultînd schema structurală din figura 5.7. Devine astfel ușor de exprimat funcția de fiabilitate sau funcția de repartiție a timpului de funcționare a sistemului pe baza funcțiilor de repartiție a timpului de funcționare corespunzătoare fiecărei unități H_i, notate în continuare $F_u(t)$, respectiv $R_u(t)$ — funcția de fiabilitate.

Pentru dezvoltarea modelului logic de fiabilitate se fac următoarele trei ipoteze:

(a) defectările unităților H_i sînt statistic independente între ele;

(b) ratele de defectare ale unităților H_i sînt constante în timp și egale între ele;

(c) pentru codarea stărilor sistemului secvențial se folosește un cod corector de l erori.

Cu aceste considerente se poate scrie expresia funcției de repartiție corespunzătoare acestei structuri fiabilistice:

$$F_{SR}(t) = \sum_{i=l+1}^{n} C_n^i F_u(t)^i R_u(t)^{n-i} \tag{5.20}$$

În condițiile ipotezei (b) rezultă:

$$F_{SR}(t) = \sum_{i=l+1}^{n} C_n^i (1 - e^{-\lambda_u t})^i \, e^{-(n-i)\lambda_u t} \tag{5.21}$$

adică sistemul se va defecta atunci cînd se vor defecta mai mult de l unități H_i, celelalte unități funcționînd corect. În relațiile (5.20) și (5.21) indicii „u" se referă la unitățile H_i, identice din punctul de vedere al fiabilității.

Considerîndu-se ratele de defectare egale, este posibilă — în primul rînd — o analiză calitativă a performanțelor de fiabilitate ale structurii.

În general însă pentru că sînt de complexități diferite, unitățile H_i nu au rate de defectare identice. De aceea, în vederea unei analize de fiabilitate

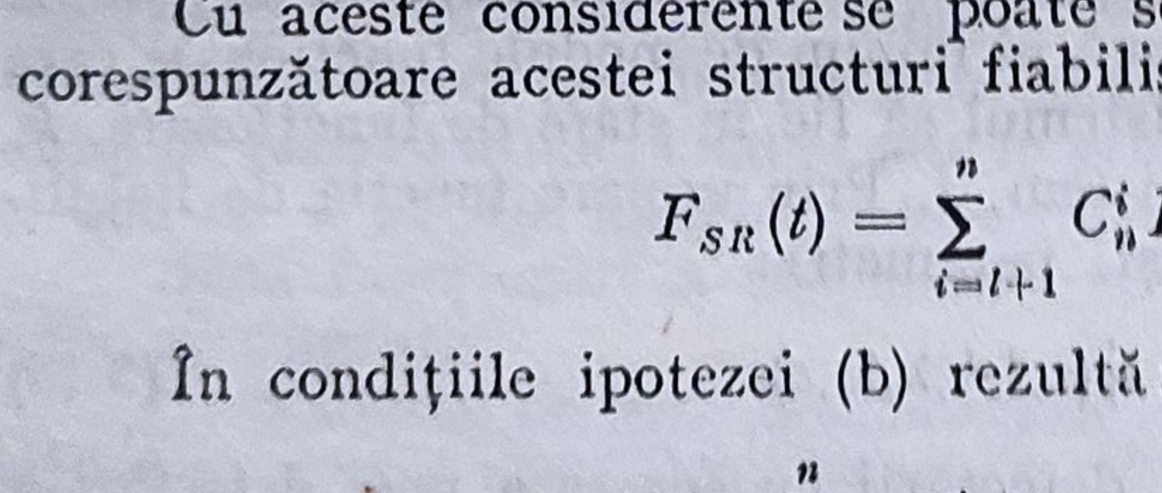


Fig. 5.7. Schema echivalentă a unui automat cu stări redondante.

mai detaliată se introduce funcţia de complexitate C_i, ca fiind numărul elementelor virtuale, independente, de aceeaşi complexitate, constituente ale unităţilor H_i şi care caracterizează complexităţile diferite ale acestora. De exemplu, dacă unitatea H_i este un circuit integrat, atunci C_i va avea o variaţie liniară cu numărul conexiunilor intrare-ieşire [39].

Într-o analiză simplificatoare, avută în vedere în continuare, funcţiile de complexitate C_i se consideră identice pentru fiecare unitate H_i, ceea ce înseamnă că se menţine în continuare ipoteza (b).

$$\lambda_{ut} = \lambda_u,$$

iar

$$\lambda_u = C\lambda. \tag{5.22}$$

Înlocuind λ_u cu $C\lambda$ în ecuaţia (5.21), se obţine:

$$F_{SR}(t) = \sum_{i=l+1}^{n} C_n^i (1 - e^{-C\lambda t})^i \, e^{-C(n-i)\lambda t} \tag{5.23}$$

Evident, funcţia de complexitate C — crescătoare o dată cu n şi l — — depinde de codul folosit, de funcţiile logice implementate, precum şi de eficienţa tehnicilor de minimizare.

Ecuaţia (5.23) permite studierea performanţelor de fiabilitate ale sistemului analizat în funcţie de n şi l [7]. Astfel, se poate identifica un cod eficient care, utilizat în acest mod, să conducă la un sistem cu performanţe de fiabilitate maximă.

Dacă $C\lambda t << 1$, atunci (5.23) devine:

$$F_{SR}(t) = \sum_{i=l+1}^{n} C_n^i \, (C\lambda t)^i \, (1 - C\lambda t)^{n-i} \tag{5.24}$$

care se aproximează asimptotic cu ecuaţia:

$$F_{SR}(t) = C_n^{l+1} (C\lambda t)^{l+1} \frac{1-(n-l-1)C\lambda t}{1-\frac{n-1}{l+1}\cdot\frac{C\lambda t}{1-C\lambda t}} \tag{5.25}$$

Dacă $nC\lambda t \ll 1$, atunci (5.24) se aproximează în continuare cu:

$$F_{SR}(t) = C_n^{l+1} (C\lambda t)^{l+1} \tag{5.26}$$

Prin logaritmarea ecuaţiei (5.26), se obţine:

$$\log F_{SR}(t) = \log C_n^{l+1} + (l+1) \log C\lambda t, \tag{5.27}$$

expresie care permite analiza grafică a variaţiei funcţiei de repartiţie a timpului de funcţionare cu creşterea gradului redondanţei. În figura 5.8 este trasat graficul funcţiei $\log F_{SR}(t)$, avînd variabilă $\log C\lambda t$, iar parametri n şi l. Se observă imediat că funcţia de repartiţie a timpului de funcţionare a sistemului, care are semnificaţia probabilităţii de defectare a sistemului, are o variaţie descrescătoare cu creşterea gradului redondanţei; rezultatul

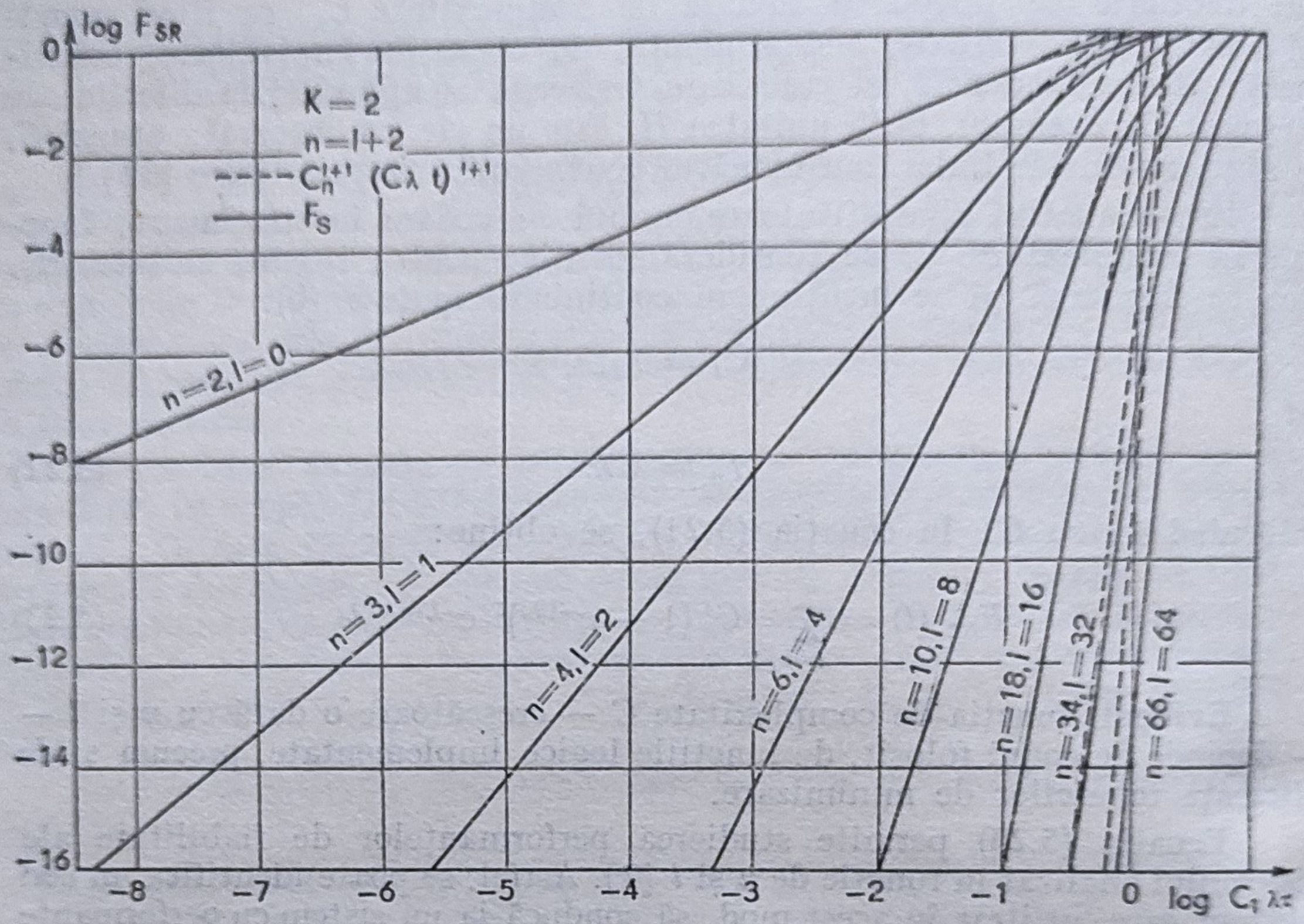


Fig. 5.8. Variația funcției de repartiție a timpului de funcționare a unui automat cu stări redondante în cazul în care factorul de complexitate C este constant.

anterior este obținut în condițiile acceptării ipotezei simplificatoare în conformitate cu care funcția de complexitate C este constantă la variația lui n. De fapt, funcția de complexitate C nu este constantă cînd n variază, ci depinde de n, deoarece unitatea H_i devine mai complexă cu creșterea lui n.

Pentru facilitatea dezvoltărilor analitice următoare se admite că factorul de complexitate C are o dependență liniară de forma [39]:

$$C = an. \tag{5.27}$$

Cu această ipoteză ecuația (5.26) devine:

$$\log F_{SR}(t) = \log C_n^{l+1} + (l+1) \log n + (l+1) \log a\lambda t. \tag{5.28}$$

În figura 5.9 este reprezentat — pentru diferite valori ale lui n — graficul funcției $\log F_{SR}(t)$, avînd ca variabilă $\log a\lambda t$. Examinîndu-se variația funcției $\log F_{SR}(t)$ cu n, se observă că dacă $\log a\lambda t = -1{,}6$, atît pentru $n = 18$ cît și pentru $n = 66$ funcția studiată are aceeași valoare, $\log F_{SR}(t) = -6$. Se poate deci concluziona că atunci cînd n crește de la valoarea inițială $n = k$, funcția $\log F_{SR}(t)$ trece printr-un extrem (minim) pentru un anumit n, conform teoremei lui Rolle.

Considerînd $l = n - k$, în figura 5.10 este reprezentat graficul funcției $\log F_{SR}(t)$ — dată de (5.28) — cu variația lui n, pentru un anumit $a\lambda t$. Pe grafic se evidențiază minimul funcției menționate cînd $n = n_{min}$.

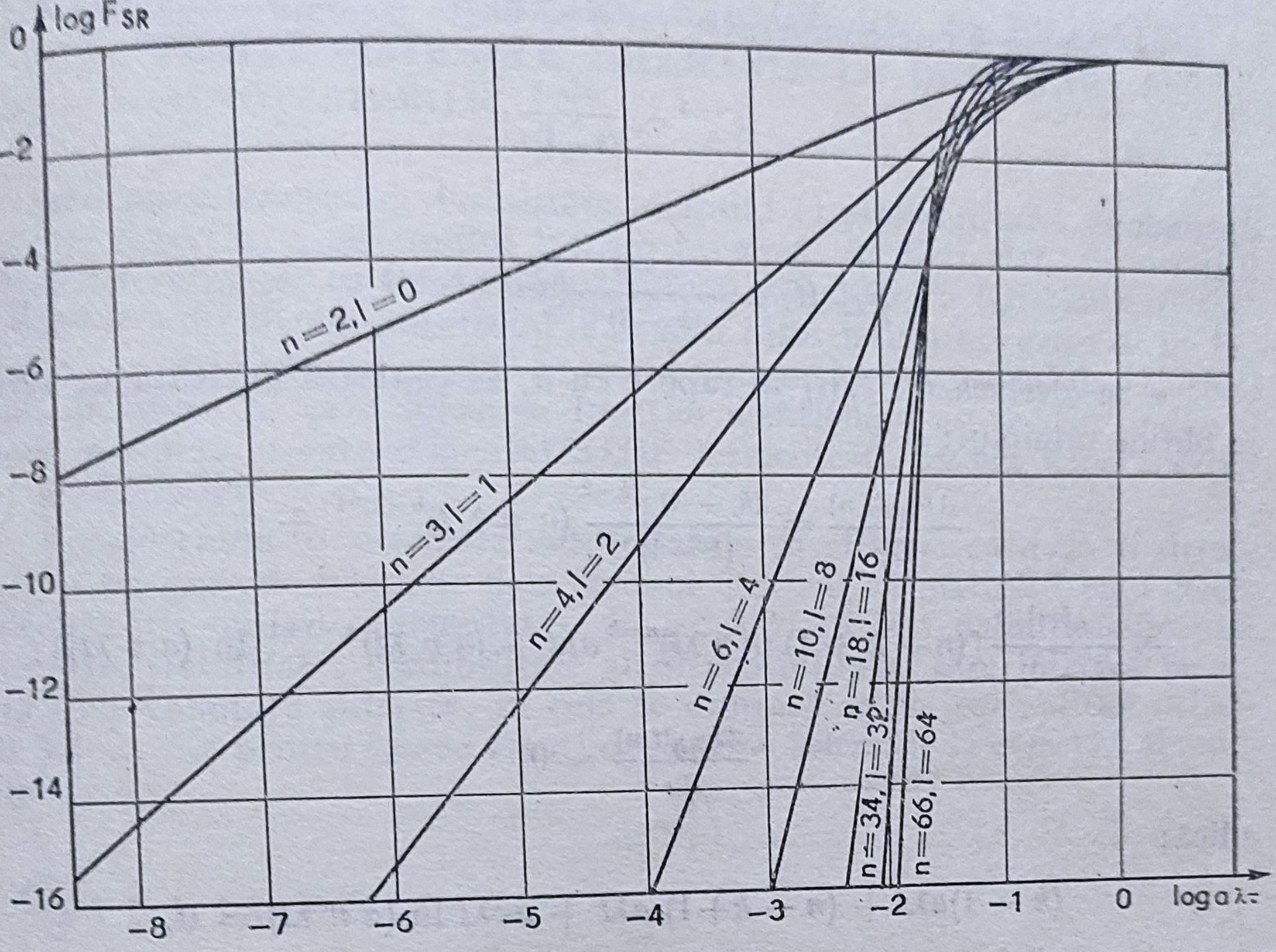


Fig. 5.9. Variația funcției de repartiție a timpului de funcționare a unui automat cu stări redondante în cazul $C = an$.

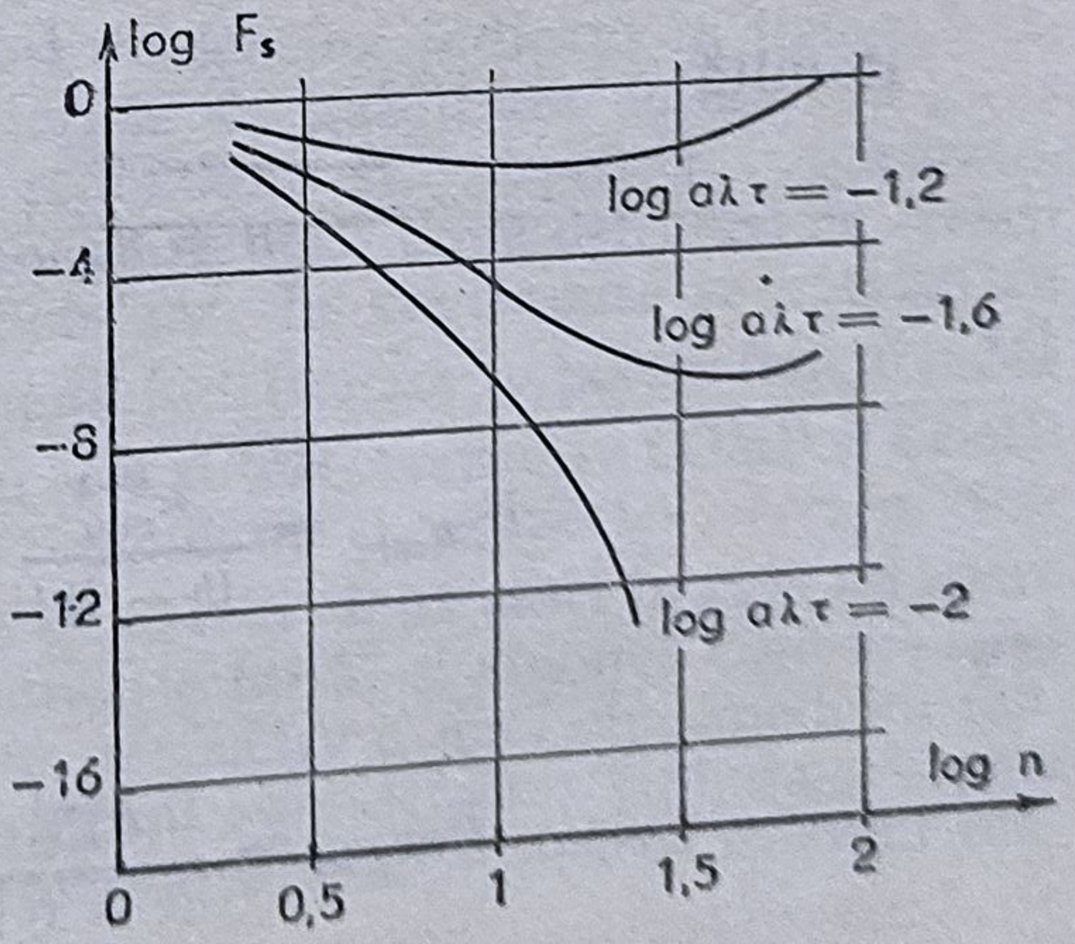


Fig. 5.10. Evidențierea minimului funcției de repartiție a timpului de funcționare a unui automat cu stări redondante la creșterea gradului redondanței.

În continuare, pentru identificarea minimului funcției se urmărește o procedură tipică din analiza matematică:

• C_n^{l+1} se înlocuiește cu:

$$C_n^{l+1} = C_n^{n-k+1} = \frac{n(n-1)\ldots(n-k+2)(n-k+1)!}{(k-1)!\,(n-k+1)!} \tag{5.29}$$

• Pentru k mic comparativ cu n se poate face aproximația:

$$C_n^{l+1} \simeq \frac{n^{k-1}}{(k-1)!},$$

de unde:

$$F_{SR}(t) \simeq \frac{n^{k-1}}{(k-1)!}(an\lambda t)^{n-k+1} \tag{5.30}$$

• Se derivează $F_{SR}(t)$ în raport cu n, se egalează derivata cu zero și se obține minimul:

$$\frac{\mathrm{d}F_{SR}(t,n)}{\mathrm{d}n} = \frac{(k-1)\,n^{k-2}}{(k-1)!}(a\,n\,\lambda\,t)^{n-k+1} +$$

$$+\frac{n^{k-1}}{(k-1)!}[(n-k+1)\,(a\,n\,\lambda t)^{n-k}\,a\lambda t + (a\,n\,\lambda t)^{n-k+1}\,\ln\,(a\,n\,\lambda t)];$$

$$\frac{\mathrm{d}F_{SR}(t,n)}{\mathrm{d}n} = 0,$$

adică:

$$(k-1)a\lambda t + (n-k+1)\,a\lambda t + an\lambda t\,\ln\,(a\,n\,\lambda t) = 0,$$

de unde se obține:

$$\ln\,an\lambda t = -1.$$

Rezultă:

$$n = n_{\min} = \frac{1}{a\lambda t\mathrm{e}}, \tag{5.31}$$

iar

$$F_{SR\min} = \frac{n_{\min}^{k-1}}{(k-1)!}\,(an_{\min}\,\lambda t)^{n_{\min}-k+1} \tag{5.32}$$

sau

$$F_{SR\min} = \frac{n_{\min}^{k-1}}{(k-1)!}\left(\frac{1}{\mathrm{e}}\right)^{n_{\min}-k+1}.$$

Analiza întreprinsă arată că, datorită creșterii complexității sistemului, probabilitatea de defectare a acestuia are o valoare limită cînd gradul redondanței se mărește.

În [7] a fost efectuată o analiză numerică a funcției de reparație a timpului de funcționare pentru această structură, luîndu-se în considerare diferite tipuri de codări și forme de dependență ale funcției de complexitate C de n; cu acest prilej s-a evidențiat existența unui minim al lui F_{SR} pentru un anumit n.

5.3.6. EVALUAREA PERFORMANȚELOR DE FIABILITATE PENTRU SISTEMELE TOLERANTE LA DEFECTĂRI ȘI AUTOTESTABILE

Utilizarea funcției de fiabilitate, definită ca probabilitatea de funcționare fără defectări a sistemului într-un interval de timp dat, în condiții apriori determinate, ca măsură cantitativă principală a funcționării fără defectări a unui sistem autotestabil ca și a altor indicatori generali de fiabilitate, nu este semnificativă în cazul sistemelor autotestabile. Motivul constă în aceea că prin folosirea funcției menționate nu se dau indicații asupra modului în care semnalul STOP influențează asupra bunei funcționări a sistemului.

Considerîndu-se sistemele autotestabile ca sisteme cu două tipuri de ieșiri — un grup de ieșiri de date și unul destinat semnalizării unui defect și declanșării opririi sistemului — în § 5.2.2. au fost definiți indicatorii securitate, fiabilitate, credibilitate și disponibilitate, care înlătură inconvenientele enunțate anterior. În cele ce urmează se vor exemplifica definițiile 5.1 ... 5.4 pentru cîteva tipuri de sisteme tolerante la defectări și autotestabile.

STUDII DE CAZ

I. Sistem autotestabil cu dublare

Se consideră sistemul autotestabil cu schema indicată în figura 5.11. Cu $R_1(t)$, $R_2(t)$ și $R_c(t)$ s-au notat funcțiile de fiabilitate definite în „sens clasic" — ale modulelor funcțional, redondant și, respectiv, comparatorului. Se admite ipoteza neapariției concomitente a două defectări de același tip în modulul funcțional și în cel redondant. În continuare se notează cu MF și $\overline{\text{MF}}$ evenimentele reprezentînd buna funcționare respectiv funcționarea defectuoasă a modulului funcțional, cu MR și $\overline{\text{MR}}$, respectiv COMP și $\overline{\text{COMP}}$ aceleași evenimente pentru modulul redondant și, respectiv, comparator.

O analiză a schemei evidențiază că există două stări în care nu apare semnalul STOP și se transmite o informație falsă, și anume: cînd modulul funcțional și comparatorul sînt defecte, respectiv atunci cînd modulul funcțional, modulul redondant și comparatorul sînt defecte. Ținîndu-se seama de raționamentul anterior, rezultă următoarea relație pentru calculul funcției de securitate:

$$\begin{aligned} \mathcal{S}(t) &= 1 - [\Pr(\overline{\text{MF}} \cap \text{MR} \cap \overline{\text{COMP}}) + \Pr(\overline{\text{MF}} \cap \overline{\text{MR}} \cap \overline{\text{COMP}})] = \\ &= 1 - R_2(t)\,(1 - R_1(t))\,(1 - R_c(t)) - (1 - R_1(t))\,((1 - R_2(t))\,((1 - R_c(t)) = \\ &= R_1(t) + R_c(t) - R_1(t)R_c(t). \end{aligned} \tag{5.33}$$

Analizînd sistemul considerat în scopul calculării funcției de fiabilitate se constată că nu există decît o stare în care acesta este disponibil — semnalul STOP nu a apărut — și informația dată la ieșirea sistemului

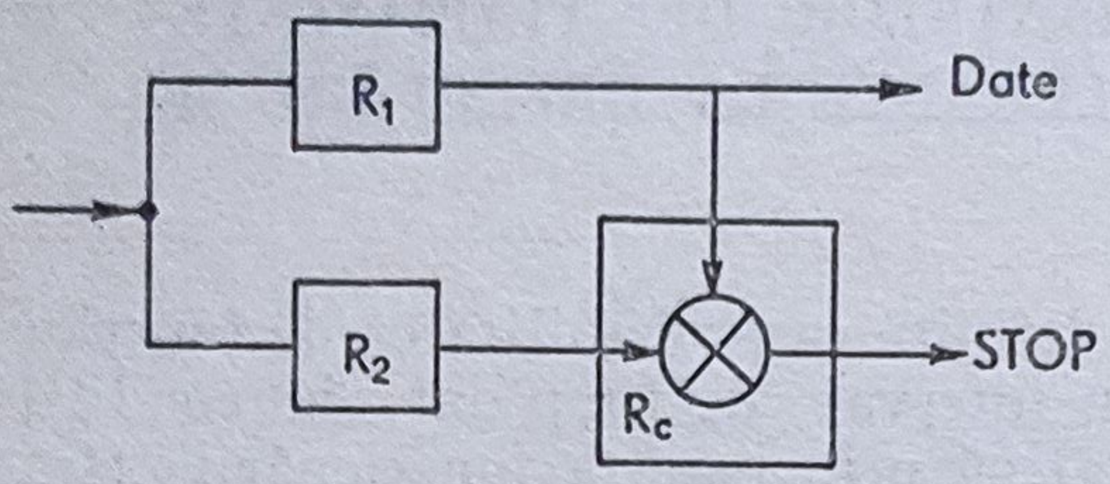


Fig. 5.11. Modelul structural al unui sistem autotestabil cu dublare.

este corectă, și anume atunci cînd modulul funcțional, modulul redondant și comparatorul sînt în stare de bună funcționare. Deci funcția de fiabilitate a sistemului este de forma:

$$R_{SA}(t) = \Pr(\mathrm{MF} \cap \mathrm{MR} \cap \mathrm{COMP}) = R_1(t)R_2(t)R_c(t). \quad (5.34)$$

Pentru a facilita calculul funcției de credibilitate, se observă că stările în care informația este corectă sînt $\mathrm{MF} \cap \mathrm{MR} \cap \mathrm{COMP}$ sau $\mathrm{MF} \cap \mathrm{MR} \cap \overline{\mathrm{COMP}}$ sau $\mathrm{MF} \cap \overline{\mathrm{MR}} \cap \overline{\mathrm{COMP}}$, iar stările sistemului în care acesta este disponibil — nu a apărut semnalul STOP — sînt: $\overline{\mathrm{MF}} \cap \mathrm{MR} \cap \mathrm{COMP}$ sau $\mathrm{MF} \cap \overline{\mathrm{MR}} \cap \overline{\mathrm{COMP}}$ sau $\overline{\mathrm{MF}} \cap \mathrm{MR} \cap \overline{\mathrm{COMP}}$ sau $\overline{\mathrm{MF}} \cap \overline{\mathrm{MR}} \cap \overline{\mathrm{COMP}}$.

Conform definiției date credibilitatea se poate scrie

$$\mathscr{C}(t) = \frac{\Pr(\mathrm{MF} \cap \mathrm{MR} \cap \mathrm{COMP}) + \Pr(\mathrm{MF} \cap \overline{\mathrm{MR}} \cap \overline{\mathrm{COMP}}) + \Pr(\mathrm{MF} \cap \mathrm{MR} \cap \overline{\mathrm{COMP}})}{\Pr(\mathrm{MF} \cap \mathrm{MR} \cap \mathrm{COMP}) + \Pr(\mathrm{MF} \cap \mathrm{MR} \cap \overline{\mathrm{COMP}}) + \Pr(\mathrm{MF} \cap \overline{\mathrm{MR}} \cap \overline{\mathrm{COMP}}) + \Pr(\overline{\mathrm{MF}} \cap \mathrm{MR} \cap \overline{\mathrm{COMP}}) + \Pr(\overline{\mathrm{MF}} \cap \overline{\mathrm{MR}} \cap \overline{\mathrm{COMP}}) =$$

$$= \frac{R_1(t)R_2(t)R_c(t) + R_1(t)R_2(t)(1 - R_c(t)) + R_1(t)(1 - R_2(t))(1 - R_c(t))}{R_1(t)R_2(t)R_c(t) + R_1(t)R_2(t)(1 - R_c(t)) + R_1(t)(1 - R_2(t))(1 - R_c(t)) + R_2(t)(1 - R_1(t))(1 - R_c(t)) + (1 - R_1(t))(1 - R_2(t))(1 - R_c(t))} \quad (5.35)$$

Dacă modulul funcțional este identic cu cel de rezervă — situației uzuală — relațiile (5.33), (5.34) și (5.35) devin:

$$\mathscr{S}(t) = R(t)\,(1 - R_c(t)) + R_c(t); \quad (5.36)$$

$$R_{SA}(t) = R(t)^2 R_c(t); \quad (5.37)$$

$$\mathscr{C}(t) = \frac{R(t)^2 R_c(t) + R(t)^2(1 - R_c(t)) + R(t)(1 - R(t))(1 - R_c(t))}{R(t)^2 R_c(t) + R(t)^2(1 - R_c(t)) + 2R(t)(1 - R(t))(1 - R_c(t))} = \frac{R(t)^2 + R(t)(1 - R(t))(1 - R_c(t))}{R(t)^2 + (1 - R_c(t))(1 - R(t)^2)}. \quad (5.38)$$

Exemplul numeric următor evidențiază că pentru un sistem autotestabil — chiar atunci cînd funcția de fiabilitate are o valoare relativ mică — există puține șanse de a se transmite o informație falsă nedetectată.

Exemplul 5.2. Pentru un timp al misiunii T_m, se consideră că valorile funcțiilor de fiabilitate ale elementelor componente ale unui sistem autotestabil cu dublare — și anume modulul funcțional, modulul redondant și comparatorul — sînt:

$$R_1(T_m) = R_2(T_m) = 0{,}8 \text{ și } R_c(T_m) = 0{,}9.$$

Utilizîndu-se relaţiile (5.36) (5.37) şi (5.38), se pot calcula indicatorii specifici $\mathcal{S}(T_m) = 0{,}98$, $R_{SA}(T_m) = 0{,}58$ şi $\mathcal{C}(T_m) = 0{,}97$ ale căror valori numerice susţin afirmaţia de mai sus.

II. Sistem autotestabil redondant

Se consideră sistemul autotestabil avînd structura redondantă indicată în figura 2.6*b*, sistem care asigură o disponibilitate mai mare decît structura analizată anterior.

Dacă apare un dezacord între cele două unităţi ale aceluiaşi grup — — de bază sau rezervă — informaţia este blocată la ieşire. În cazul cînd cele două unităţi ale aceluiaşi grup se defectează, defectele respective se pot traduce printr-o aceeaşi ieşire falsă şi comparatorul nu va mai bloca trecerea acestei informaţii false.

Structura schemei analizate în vederea calculului indicatorilor de fiabilitate este indicată în figura 5.12; cu $R_{c1}(t)$ şi $R_{c2}(t)$ s-au notat funcţiile de fiabilitate ale celor două comparatoare şi cu $R_{MULT}(t)$ — funcţia de fiabilitate a multiplexorului.

Calculul funcţiei de securitate. Mai întîi este necesară stabilirea stărilor „nesigure“ ale sistemului. Din analiza funcţionării schemei rezultă că aceste stări sînt următoarele:

- $\overline{\text{MULT}}$ — multiplexorul este defect — stare nesigură, oricare ar fi stările celorlalte componente, deoarece în acest caz ieşirile de date şi ieşirile STOP sînt funcţie de stările componentelor care concură la realizarea lor;
- $\text{MULT} \cap \overline{C}_1 \cap \overline{C}_2$: dacă cele patru unităţi funcţionale sînt în stare de bună funcţionare nu se pune problema unei ieşiri false nedetectate; în cazul în care comutatoarele C_1 şi C_2 sînt blocate în „1“, datele de ieşire pot fi false, dar se va produce semnalul STOP de vreme ce multiplexorul funcţionează corect. Dacă o unitate funcţională este defectă, celelalte trei unităţi fiind în stare de bună funcţionare, apare ca o stare nesigură cînd grupul de comutatoare C_1 şi C_2 este blocat în „0“ respectiv „1“ etc.

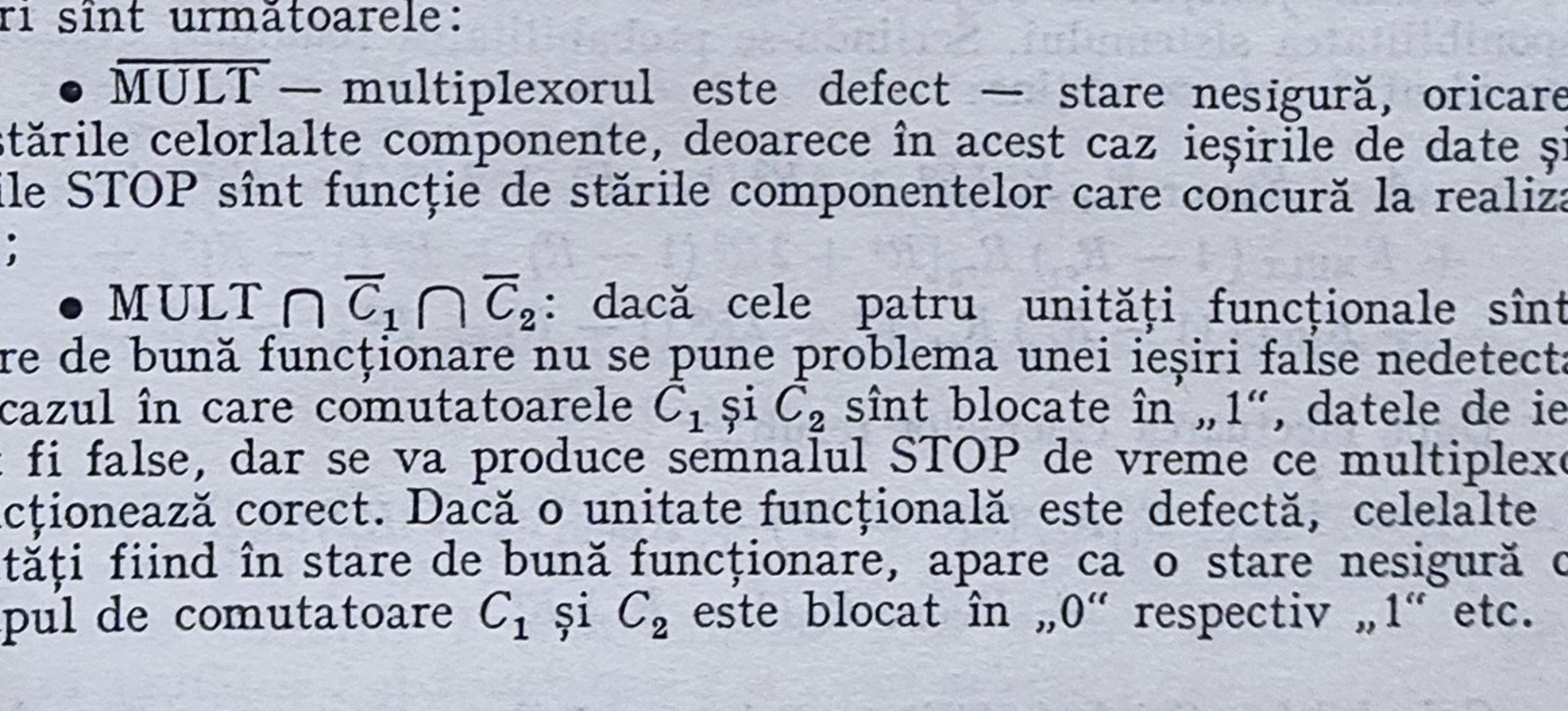


Fig. 5.12. Structura unui sistem autotestabil redondant.

Examinîndu-se în aceeași manieră toate stările posibile, se poate scrie:

$$1 - \mathcal{S} = 1 - R_{\text{MULT}} + R_{\text{MULT}}\,(1 - R_{c_1})\,(1 - R_{c_2})[2R^3(1 - R) + 5R^2\,(1 - R)^2 + 4R\,(1 - R)^3 + (1 - R)^4] + R_{\text{MULT}}\,(1 - R_{c_1})\,R_{c_2}[R^3(1 - R) + 4R^2\,(1 - R)^2 + 4R\,(1 - R)^3 + (1 - R)^4] + R_{\text{MULT}}\,(1 - R_{c_2})\,R_{c_1}[R^3(1 - R) + 4R^2\,(1 - R)^2 + 4R\,(1 - R)^3 + (1 - R)^4] + R_{\text{MULT}}\,R_{c_1}\,R_{c_2}[2R^2\,(1 - R)^2 + 4R\,(1 - R)^3 + (1 - R)^4].$$

Pentru simplificarea scrierii, nu s-a mai explicitat dependența de timp a funcțiilor R și $\mathcal{S}$.

Această ecuație prelucrată conduce la următoarea expresie a funcției de securitate:

$$\mathcal{S} = R_{\text{MULT}}[1 - (1 - R_{c_1})\,(1 - R_{c_2})\,(1 - R^2) - R_{c_1}(1 - R_{c_2})\,(1 - 2R^2 + R^3) - R_{c_2}(1 - R_{c_1})(1 - 2R^2 + R^3) - R_{c_1}R_{c_2}(1 - 4R^2 + 4R^3 - R^4)]. \tag{5.39}$$

Calculul funcției de disponibilitate. Se identifică stările care asigur disponibilitatea sistemului. Scriindu-se probabilitatea reuniunii evenimentelor identificate, se obține:

$$A = R_{\text{MULT}}\,R_{c_1}\,R_{c_2}[R^4 + 4R^3\,(1 - R) + 2R^2\,(1 - R)^2] + R_{\text{MULT}}\,(1 - R_{c_1})\,R_{c_2}[R^4 + 2R^3\,(1 - R) + R^2\,(1 - R)^2] + R_{\text{MULT}}\,(1 - R_{c_2})\,R_{c_1}[R^4 + 2R^3\,(1 - R) + R^2\,(1 - R)^2].$$

După efectuarea calculelor rezultă:

$$A = R_{\text{MULT}}\,R_{c_1}R_{c_2}\,(2R^2 - R^4) + R_{\text{MULT}}\,(1 - R_{c_1})\,R_{c_2}R^2 + R_{\text{MULT}}\,R_{c_1}(1 - R_{c_2})\,R^2. \tag{5.40}$$

Calculul funcției de fiabilitate. Printre stările pentru care sistemul este disponibil există unele în care semnalele de date pot fi false. După eliminarea acestor stări se poate — ținîndu-se seama de stările de disponibilitate — să se scrie expresia funcției de fiabilitate a acestuia (în sensul definiției 5.2):

$$R_{SA} = R_{\text{MULT}}\;R_{c_1}R_{c_2}[R^4 + 4R^3\,(1 - R)] + R_{\text{MULT}}\,(1 - R_{c_1})\,R_{c_2}\,[R^4 + R^3\,(1 - R)] + R_{\text{MULT}}\,(1 - R_{c_2})\,R_{c_1}\,[R^4 + R^3\,(1 - R)]$$

sau

$$R_{SA} = R_{\text{MULT}}\;R_{c_1}R_{c_2}\,R^3\,(4 - 3R) + R_{\text{MULT}}\,(1 - R_{c_1})\,R_{c_2}\,R^3 + R_{\text{MULT}}\,R_{c_1}\,(1 - R_{c_2})\,R^3. \tag{5.41}$$

Calculul funcției de credibilitate. Conform definiției 5.4 se poate scrie:

$$\mathcal{C} = \frac{\text{Pr (ieșiri corecte} \cap \text{sistem disponibil)}}{\text{Pr (sistem disponibil)}}.$$

O analiză a schemei funcționale evidențiază toate stările care fac ca sistemul să fie disponibil. Se notează cu Y evenimentul „sistemul este disponibil"; acest eveniment este dat de relația:

$$Y = \overline{\text{MULT}} \cup \text{MULT} \cap C_1 \cap C_2 \cap [\text{„4 unități funcționează"} \cup \cup \text{„3 unități din 4 funcționează"} \cup \cup \text{„2 unități din 4 funcționează"} \cup \cup \text{„1 unitate din 4 funcționează"} \cup \text{„nu funcționează nici o unitate"}] \cup \cup \text{MULT} \cap \overline{C}_1 \cap C_2 \cup \text{MULT} \cap C_1 \cap \overline{C}_2 \cup \text{MULT} \cap \overline{C}_1 \cap \overline{C}_2$$

Pornind de la această expresie, se poate scrie:

$$\begin{aligned}\Pr(Y) = {} & 1 - R_{\text{MULT}} + R_{\text{MULT}} R_{c_1} R_{c_2} [R^4 + 4R^3 (1 - R) + 2R^2 (1 - R)^2 + \\ & + 4R (1 - R)^3 + (1 - R^4)] + R_{\text{MULT}} (1 - R_{c_1}) R_{c_2} + \\ & + R_{\text{MULT}} R_{c_1} (1 - R_{c_2}) + R_{\text{MULT}} (1 - R_{c_1}) (1 - R_{c_2}). \qquad (5.42)\end{aligned}$$

Dintre stările care conduc la disponibilitatea sistemului se aleg acelea pentru care informația dată de sistem este corectă. Se notează cu X evenimentul „ieșiri corecte $\cap$ se poate dispune de sistem"; acesta este dat de relația:

$$X = \text{MULT} \cap C_1 \cap C_2 \cap (\text{„4 unități funcționează"} \cup \text{„3 unități din 4 funcționează"}) \cup$$

$$\cup \text{MULT} \cap \overline{C}_1 \cap C_2 \cap (\text{„4 unități funcționează„} \cup \text{„3 unități din 4 funcționează mai puțin unitatea } 1 \text{ defectă și celelalte funcționează*"} \cup \text{funcționează unitățile } 1 \text{ și } 4, \text{ unitățile } 2 \text{ și } 3 \text{ sînt defecte sau funcționează unitățile } 1 \text{ și } 3, \text{ unitățile } 2 \text{ și } 4 \text{ sînt defecte"}) \cup$$

$$\cup \text{MULT} \cap C_1 \cap \overline{C}_2 \cap (\text{„4 unități funcționează"} \cup \text{„3 unități din 4 funcționează mai puțin unitatea 2 defectă și celelalte funcționează"} \cup \text{„funcționează unitățile } 2 \text{ și } 3, \text{ unitățile 1 și 4 sînt defecte sau funcționează unitățile } 2 \text{ și } 4, \text{ unitățile } 1 \text{ și } 3 \text{ sînt defecte"}) \cup$$

$$\cup \text{MULT} \cap \overline{C}_1 \cap \overline{C}_2 \cap (\text{„4 unități funcționează"} \cup \text{„funcționează unitățile } 1, 3 \text{ și } 4, \text{ unitatea } 2 \text{ este defectă sau funcționează unitățile } 1, 2 \text{ și } 3, \text{ unitatea } 4 \text{ este defectă"} \cup \text{„funcționează unitățile } 1 \text{ și } 3, \text{ unitățile } 2 \text{ și } 4 \text{ sînt defecte"}).$$

---

* Dacă unitatea 1 este defectă, C_1 poate lăsa spre ieșire o valoare falsă; dacă unitatea 2 este defectă, C_2 lasă spre ieșire o valoare adevărată; dacă unitatea 3 este defectă, atunci cînd nu apare semnalul STOP, ieșirea este bună; dacă unitatea 4 este defectă, idem.

Folosindu-se raționamentul anterior, se poate scrie:

$$\begin{aligned}\Pr(X) = {} & R_{\text{MULT}}\, R_{c1} R_{c2} (4R^3 - 3R^4) + \\ & + R_{\text{MULT}}\, (1 - R_{c1})\, R_{c2}\, (2R^2 - R^3) + \\ & + R_{\text{MULT}}\, (1 - R_{c2})\, R_{c1}\, (2R^2 - R^3) + \\ & + R_{\text{MULT}}\, (1 - R_{c1})\, (1 - R_{c2})\, R^2, \end{aligned} \tag{5.43}$$

iar

$$\mathcal{C} = \frac{\Pr(X)}{\Pr(Y)}. \tag{5.44}$$

III. Sistem TMR autotestabil

În figura 5.13 se prezintă schema unui sistem TMR cu semnal STOP. Acesta din urmă apare atunci cînd se defectează două din cele trei unități funcționale. Se fac următoarele notații: R — funcțiile de fiabilitate ale unităților funcționale; R_v — funcția de fiabilitate a voterului; R_{c1}, R_{c2}, R_{c3} — — funcțiile de fiabilitate ale elementelor comparatoare; $\overline{V}$ — evenimentul „voterul este în stare de bună funcționare"; V — evenimentul „voterul este defect"; C_1, C_2, C_3 — evenimentele „comparatoarele C_1, C_2 respectiv C_3 sînt în stare bună de funcționare"; $\overline{C}_1$, $\overline{C}_2$, $\overline{C}_3$ — evenimentul „comparatoarele C_1, C_2 respectiv C_3 sînt defecte;" 1, 2, 3 — unitățile 1, 2, 3 sînt în stare bună de funcționare; $\overline{1}$, $\overline{2}$, $\overline{3}$ — unitățile corespunzătoare sînt defecte.

Analiza funcționării schemei evidențiază evenimentele care conduc la ieșiri false sau corecte (tabelul 5.2).

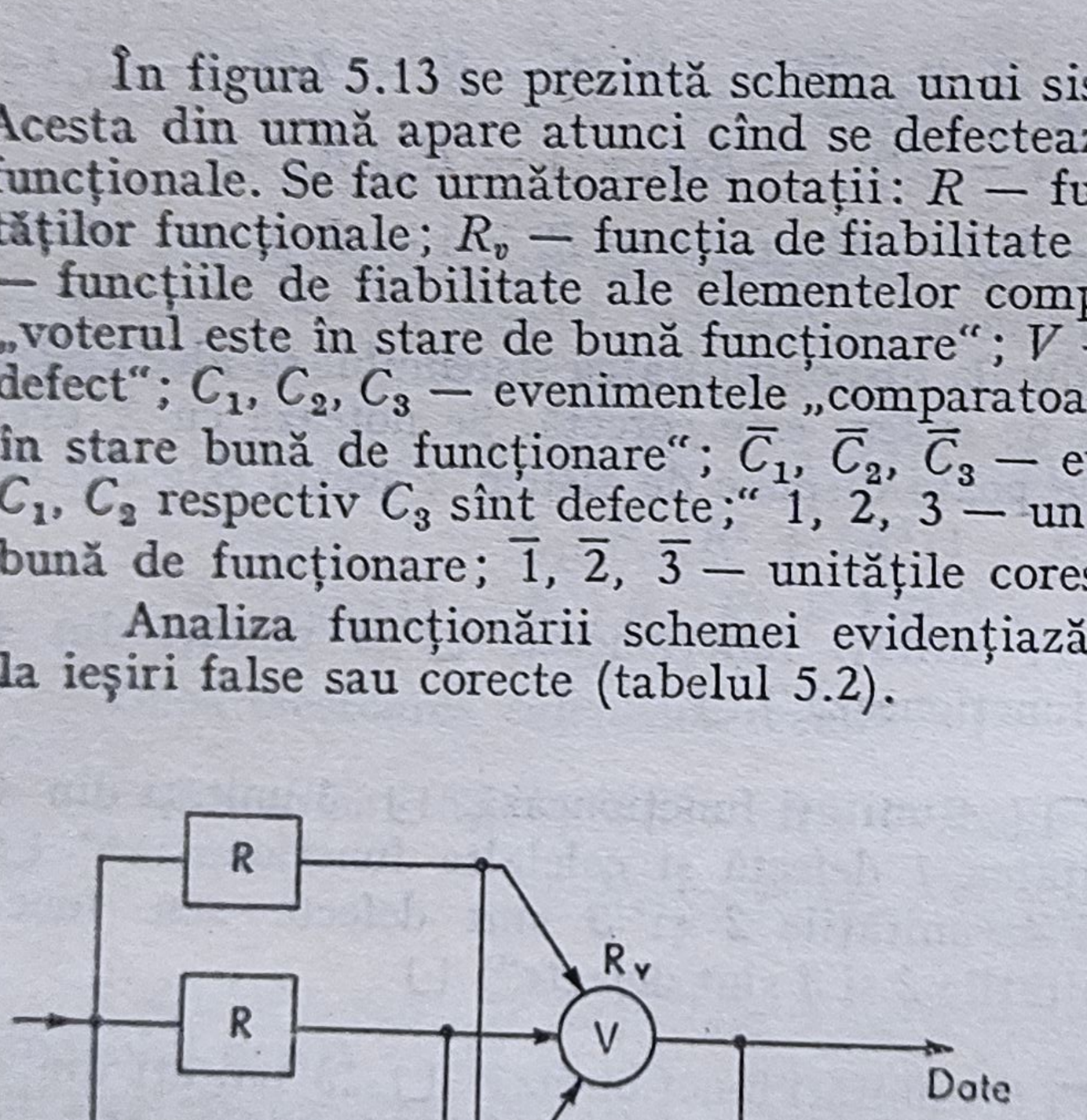


Fig. 5.13. Modelul structural al unui sistem TMR cu detector de defecte.

Tabelul 5.2

Starea sistemului	Comentarii
$\overline{V}$ —	— ieşire falsă
$V \cap C_1 \cap C_2 \cap C_3 \cap 1 \cap 2 \cap 3$	— ieşire corectă
$V \cap C_1 \cap C_2 \cap \overline{C}_3 \cap 1 \cap 2 \cap 3$	— ieşire corectă
$V \cap C_1 \cap C_2 \cap \overline{C}_3 \cap 1 \cap 2 \cap \overline{3}$	— ieşire corectă
$V \cap C_1 \cap C_2 \cap \overline{C}_3 \cap 1 \cap 2 \cap 3$	— ieşire falsă
$V \cap C_1 \cap C_2 \cap \overline{C}_3 \cap 1 \cap \overline{2} \cap \overline{3}$	— ieşire falsă
$V \cap C_1 \cap C_2 \cap \overline{\ }_3 \cap \overline{1} \cap \overline{2} \cap \overline{3}$	— ieşire falsă
$V \cap C_1 \cap \overline{C}_2 \cap C_3 \cap 1 \cap 2 \cap 3$	— ieşire corectă
$V \cap C_1 \cap \overline{C}_2 \cap C_3 \cap 1 \cap \overline{2} \cap 3$	— ieşire corectă
$V \cap C_1 \cap \overline{C}_2 \cap C_3 \cap \overline{1} \cap \overline{2} \cap 3$	— ieşire falsă
$V \cap C_1 \cap \overline{C}_2 \cap C_3 \cap 1 \cap \overline{2} \cap \overline{3}$	— ieşire falsă
$V \cap C_1 \cap \overline{C}_2 \cap C_3 \cap \overline{1} \cap \overline{2} \cap \overline{3}$	— ieşire falsă
$V \cap \overline{C}_1 \cap C_2 \cap C_3 \cap 1 \cap 2 \cap 3$	— ieşire corectă
$V \cap \overline{C}_1 \cap C_2 \cap C_3 \cap \overline{1} \cap 2 \cap 3$	— ieşire corectă
$V \cap \overline{C}_1 \cap C_2 \cap C_3 \cap \overline{1} \cap 2 \cap \overline{3}$	— ieşire falsă
$V \cap \overline{C}_1 \cap C_2 \cap C_3 \cap \overline{1} \cap \overline{2} \cap 3$	— ieşire falsă
$V \cap \overline{C}_1 \cap C_2 \cap C_3 \cap \overline{1} \cap \overline{2} \cap \overline{3}$	— ieşire falsă
$V \cap C_1 \cap \overline{C}_2 \cap \overline{C}_3 \cap 1 \cap 2 \cap 3$	— ieşire corectă
$V \cap C_1 \cap \overline{C}_2 \cap \overline{C}_3 \cap 1 \cap \overline{2} \cap 3$	— ieşire corectă
$V \cap C_1 \cap \overline{C}_2 \cap \overline{C}_3 \cap 1 \cap 2 \cap \overline{3}$	— ieşire falsă
$V \cap C_1 \cap \overline{C}_2 \cap \overline{C}_3 \cap 1 \cap \overline{2} \cap \overline{3}$	— ieşire falsă
$V \cap C_1 \cap \overline{C}_2 \cap \overline{C}_3 \cap \overline{1} \cap \overline{2} \cap 3$	— ieşire falsă
$V \cap C_1 \cap \overline{C}_2 \cap \overline{C}_3 \cap \overline{1} \cap 2 \cap \overline{3}$	-- ieşire falsă
$V \cap C_1 \cap \overline{C}_2 \cap \overline{C}_3 \cap \overline{1} \cap \overline{2} \cap \overline{3}$	— ieşire falsă
$V \cap \overline{C}_1 \cap C_2 \cap \overline{C}_3 \cap 1 \cap 2 \cap 3$	— ieşire corectă
$V \cap \overline{C}_1 \cap C_2 \cap \overline{C}_3 \cap \overline{1} \cap 2 \cap 3$	— ieşire corectă
$V \cap \overline{C}_1 \cap C_2 \cap \overline{C}_3 \cap 1 \cap 2 \cap \overline{3}$	— ieşire corectă
$V \cap \overline{C}_1 \cap C_2 \cap \overline{C}_3 \cap \overline{1} \cap 2 \cap \overline{3}$	— ieşire falsă
$V \cap \overline{C}_1 \cap C_2 \cap \overline{C}_3 \cap 1 \cap \overline{2} \cap 3$	— ieşire falsă
$V \cap \overline{C}_1 \cap C_2 \cap \overline{C}_3 \cap 1 \cap \overline{2} \cap \overline{3}$	— ieşire falsă
$V \cap \overline{C}_1 \cap C_2 \cap \overline{C}_3 \cap \overline{1} \cap \overline{2} \cap 3$	— ieşire falsă
$V \cap \overline{C}_1 \cap C_2 \cap \overline{C}_3 \cap \overline{1} \cap \overline{2} \cap \overline{3}$	— ieşire corectă
$V \cap \overline{C}_1 \cap \overline{C}_2 \cap C_3 \cap 1 \cap 2 \cap 3$	— ieşire corectă
$V \cap \overline{C}_1 \cap \overline{C}_2 \cap C_3 \cap \overline{1} \cap 2 \cap 3$	— ieşire corectă
$V \cap \overline{C}_1 \cap \overline{C}_2 \cap C_3 \cap 1 \cap \overline{2} \cap 3$	— ieşire corectă
$V \cap \overline{C}_1 \cap \overline{C}_2 \cap C_3 \cap \overline{1} \cap 2 \cap \overline{3}$	— ieşire falsă
$V \cap \overline{C}_1 \cap \overline{C}_2 \cap C_3 \cap \overline{1} \cap \overline{2} \cap 3$	— ieşire falsă
$V \cap \overline{C}_1 \cap \overline{C}_2 \cap C_3 \cap 1 \cap \overline{2} \cap \overline{3}$	— ieşire falsă
$V \cap \overline{C}_1 \cap \overline{C}_2 \cap C_3 \cap \overline{1} \cap \overline{2} \cap \overline{3}$	— ieşire falsă

Tabelul 5.2. (continuare)

Starea sistemului		Comentarii
$V \cap \bar{C}_1 \cap \bar{C}_2 \cap \bar{C}_3 \cap$	(2 unități din 3 funcționează bine)	— ieșire corectă
$V \cap \bar{C}_1 \cap \bar{C}_2 \cap \bar{C}_3 \cap$	(nu funcționează bine 2 unități din 3)	— ieșire falsă

Studiindu-se în același mod apariția — sau neapariția — semnalului STOP, se pot scrie relațiile pentru calculul funcțiilor securitate, fiabilitate, disponibilitate și credibilitate:

$$\mathcal{S} = R_v - R_v(1-R)[(1-R^2)(R_{c_1}R_{c_2} + R_{c_1}R_{c_3} + R_{c_2}R_{c_3} - 3R_{c_1}R_{c_2}R_{c_3}) + (1 + R - 2R^2)(1 - R_{c_1}R_{c_2} - R_{c_1}R_{c_3} - R_{c_2}R_{c_3} + 2R_{c_1}R_{c_2}R_{c_3})], \tag{5.45}$$

$$R_{\text{TMRSA}} = R_v R_{c_1} R_{c_2} R_{c_3} R^3, \tag{5.46}$$

$$A = R_{\text{TMRSA}}, \tag{5.47}$$

$$\mathcal{C} = \frac{\Pr(X)}{\Pr(Y)},$$

unde

$$\Pr(X) = R_v R_{c_1} R_{c_2} R_{c_3} R^3 + R_v R_{c_1} R_{c_2}(1-R_{c_3})R^2 + R_v R_{c_1}(1-R_{c_2})R_{c_3}R^2 + R_v R_{c_2} R_{c_3}(1-R_{c_1})R^2 + R_v R_{c_1}(1-R_{c_2})(1-R_{c_3})(2R^2-R^3) + R_v(1-R_{c_1})R_{c_2}(1-R_{c_3})(2R^2-R^3) + R(1-R_{c_1})(1-R_{c_2})R_{c_3}(2R^2-R^3) + R_v(1-R_{c_1})(1-R_{c_2})(1-R_{c_3})(3R^2-2R^3), \tag{5.48}$$

iar

$$\Pr(Y) = (1-R_v) + R_v R_{c_1} R_{c_2} R_{c_3} R^3 + R_v R_{c_1} R_{c_2}(1-R_{c_3})(1-R+R^3) + R_v R_{c_1}(1-R_{c_2})R_{c_3}(1-R+R^3) + R_v(1-R_{c_1})R_{c_2}R_{c_3}(1-R+R^3) + R_v R_{c_1}(1-R_{c_2})(1-R_{c_3})(1+R^3-R^2) + R_v(1-R_{c_1})R_{c_2}(1-R_{c_3})(1+R^3-R^2) + R_v(1-R_{c_1})(1-R_{c_2})R_{c_3}(1+R^3-R^2) + R_v(1-R_{c_1})(1-R_{c_2})(1-R_{c_3}). \tag{5.49}$$

IV. Sistem cu structură redondantă hibridă

Se consideră un sistem cu structură redondantă hibridă de tipul celui prezentat în § 2.2.7 (figura 2.22), format dintr-un sistem TMR și unul sau

două module în așteptare. În cazul unei astfel de structuri se poate implementa sau nu caracteristica de autotestabilitate. În continuare se consideră succesiv cele două cazuri.

Sistem cu structură redondantă hibridă fără semnal STOP. Schema sistemului cu structură redondantă hibridă considerat este prezentată în figura 5.14*a*. Pe schemă sînt indicate funcțiile de fiabilitate ale elementelor componente. Așa cum s-a arătat în § 5.2.2, pentru un astfel de sistem — fără semnal STOP — funcția de securitate va coincide cu funcția de fiabilitate a misiunii. Prin examinarea diverselor stări ale sistemului considerat și utilizarea aceluiași mod de lucru ca în paragrafele anterioare se determină expresia funcției de fiabilitate a sistemului în următoarele situații::

(a) *Cazul unui sistem cu un modul în așteptare*:

$$R_{SRH} = R_T\{R_{D_1}R_{D_2}R_{D_3}R_{D_4}[R^4 + 4R^3(1-R) + 6R^2(1-R)^2] + R_{D_1}R_{D_2}R_{D_3}(1-R_{D_4})[R^3 + 3R^2(1-R)] + R_{D_1}R_{D_2}(1-R_{D_3})R_{D_4}R^2 + (1-R_{D_1})R_{D_2}R_{D_3}R_{D_4}[R^3 + 3R^2(1-R)]\} \quad (5.50)$$

expresie care poate fi prelucrată pentru a fi adusă la o formă mai simplă

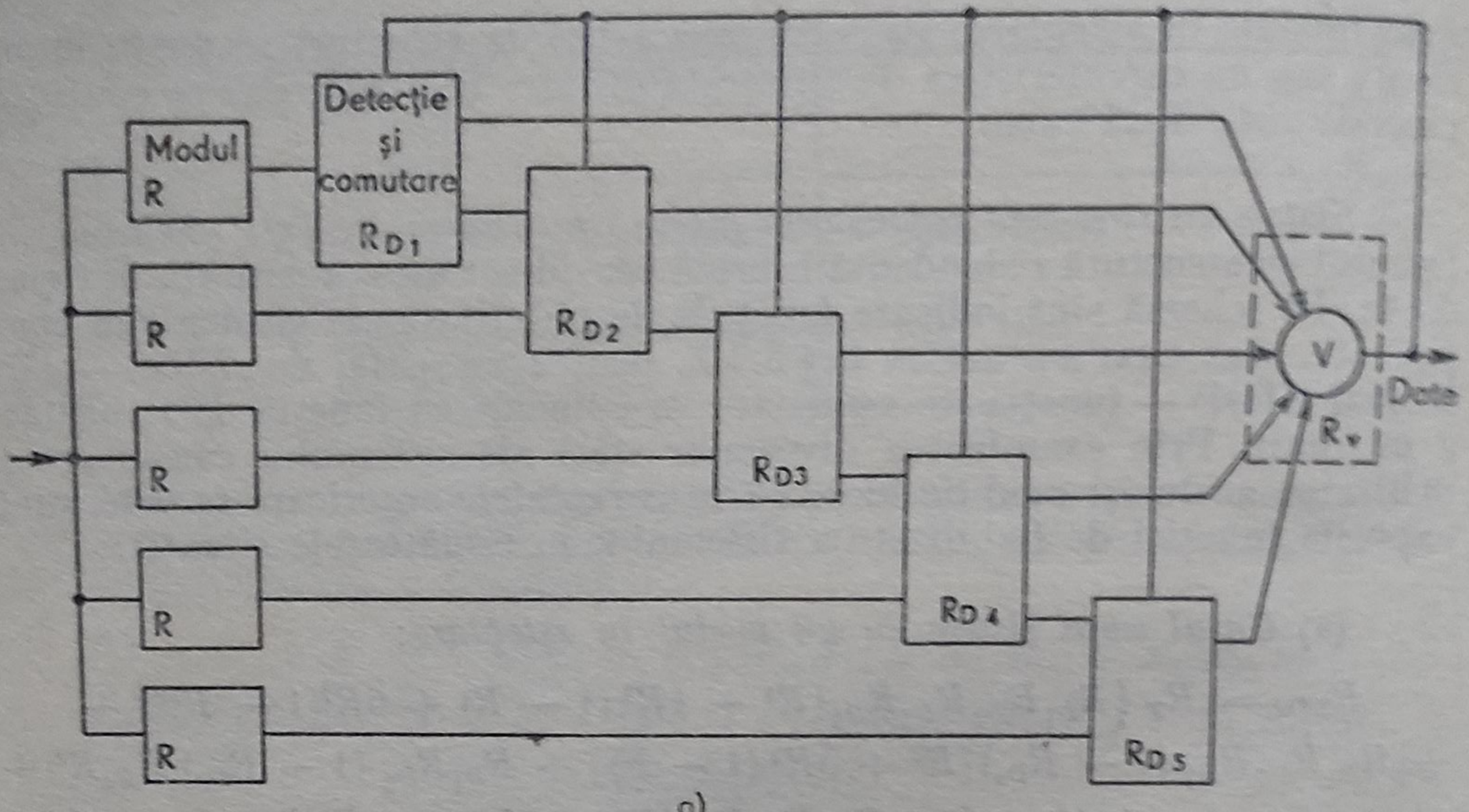
(b) *Cazul unui sistem cu două module în așteptare*:

$$R_{SRH} = R_T\{R_{D_1}R_{D_2}R_{D_3}R_{D_4}R_{D_5}[R^5 + 5R^4(1-R) + 10R^3(1-R)^2 + 10R^2(1-R)^3] + R_{D_1}R_{D_2}R_{D_3}R_{D_4}(1-R_{D_5})[R^4 + 4R^3(1-R) + 6R^2(1-R)^2] + R_{D_1}R_{D_2}R_{D_3}(1-R_{D_4})R_{D_5}[R^3 + 3R^2(1-R)] + R_{D_1}R_{D_2}(1-R_{D_3})R_{D_4}R_{D_5}R^2 + (1-R_{D_1})R_{D_2}R_{D_3}R_{D_4}R_{D_5}[R^4 + 4R^3(1-R) + 6R^2(1-R)^2]\}. \quad (5.51)$$

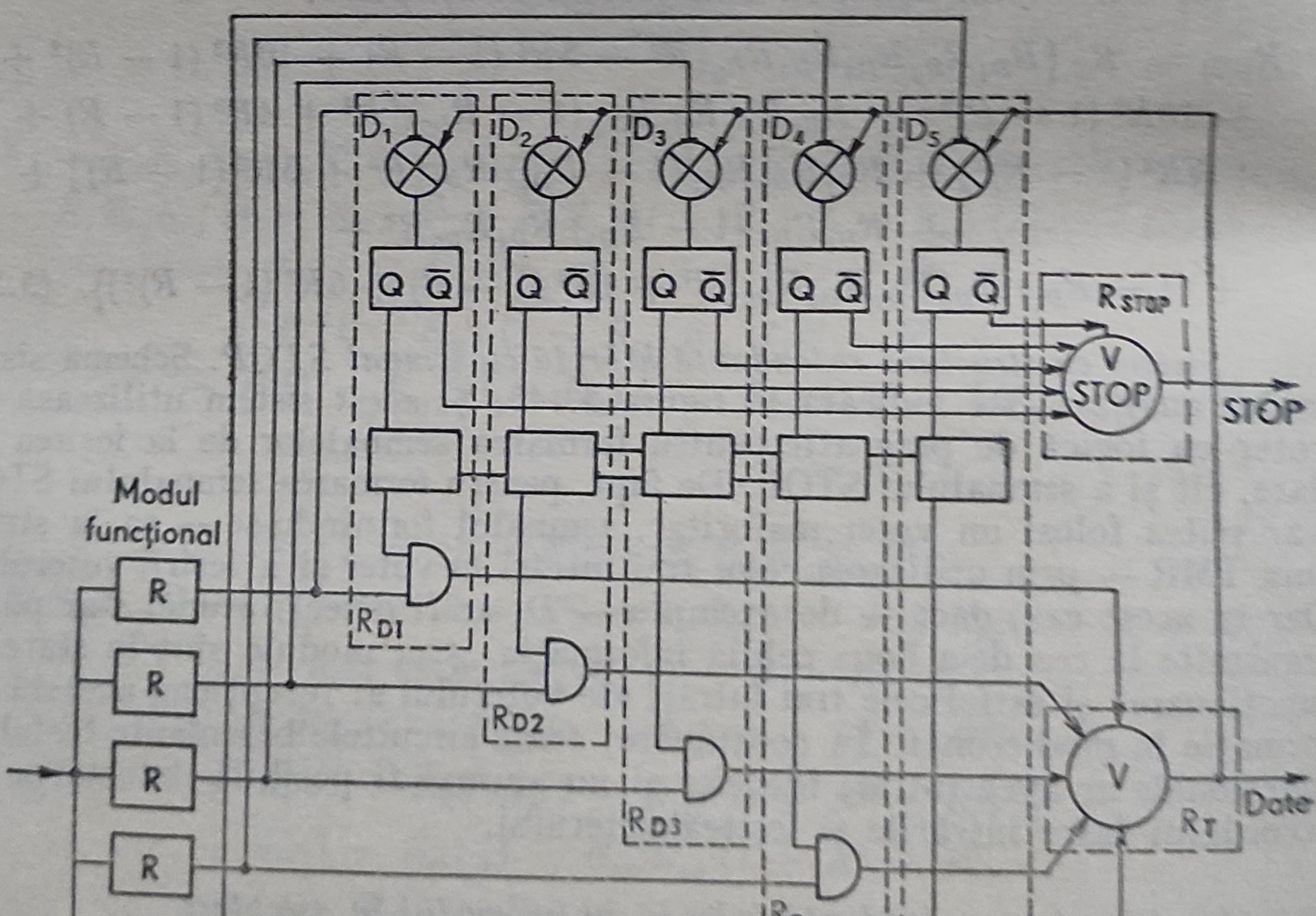
Sistem cu structură redondantă hibridă cu semnal STOP. Schema sistemului analizat este indicată în figura 5.14*b*. Și acest sistem utilizează un voter cu logică de prag atît pentru formarea semnalelor de la ieșirea de date, cît și a semnalului STOP. De fapt, pentru formarea semnalului STOP s-ar putea folosi un voter majoritar, semnalul formîndu-se — ca la structura TMR — prin utilizarea celor trei intrări în voter și a ieșirii voterului. Dar în acest caz, dacă — de exemplu — D_1 ar fi defect, atunci s-ar putea transmite la cea de-a doua celulă informația „trei module sînt în stare de funcționare" și astfel cele trei intrări ale voterului ar recepționa această informație în mod eronat. În continuare, toate circuitele basculante bistabile funcționale ar avea intrări identice și nu ar mai fi posibilă detectarea dezacordului între intrările și ieșirea voterului.

(a) *Structura redondantă hibridă cu un modul în așteptare*

Calculul funcției de securitate. Se identifică mai întîi stările nesigure ale sistemului analizat, stări indicate în tabelul 5.3. Este utilizat același mod de notare ca în paragrafele anterioare, și anume se notează cu litere mari stările de bună funcționare ale elementelor sistemului, iar cu litere mari barate — stările de defectare ale elementelor sistemului.



a)



b)

Fig. 5.14. Schema unui sistem cu structura redondantă hibridă:
a — sistem hibrid fără semnal STOP; *b* — sistem hibrid cu semnal STOP.

Tabelul 5.3

Starea sistemului	Comentarii
$\overline{\text{STOP}}$	— intervin stările nesigure ale sistemului fără semnal STOP
$\text{STOP} \cap D_1 \cap D_2 \cap D_3 \cap D_4$	— nici o stare nesigură
$\text{STOP} \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap 1 \cap 2 \cap 3 \cap 4$	— stare corectă
$\text{STOP} \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap \overline{1} \cap 2 \cap 3 \cap 4$	— se oprește dacă apare dezacord cu 2 și 3
$\text{STOP} \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap 1 \cap \overline{2} \cap 3 \cap 4$	— se oprește dacă apare dezacord cu 1 și 3
$\text{STOP} \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap 1 \cap 2 \cap \overline{3} \cap 4$	— se oprește dacă apare dezacord cu 1
$\text{STOP} \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap 1 \cap 2 \cap 3 \cap \overline{4}$	— se oprește dacă apare dezacord cu 1, 2 și 3
$\text{STOP} \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap \overline{1} \cap \overline{2} \cap 3 \cap 4$	— se oprește dacă apare dezacord cu 3
$\text{STOP} \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap \overline{1} \cap 2 \cap \overline{3} \cap 4$	— se oprește dacă apare dezacord cu 2
$\text{STOP} \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap \overline{1} \cap 2 \cap 3 \cap \overline{4}$	— se oprește dacă apare dezacord cu 2 și 3
$\text{STOP} \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap 1 \cap \overline{2} \cap \overline{3} \cap 4$	— se oprește dacă apare dezacord cu 1
$\text{STOP} \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap 1 \cap \overline{2} \cap 3 \cap \overline{4}$	— se oprește dacă apare dezacord cu 1 și 3
$\text{STOP} \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap 1 \cap 2 \cap \overline{3} \cap \overline{4}$	— se oprește dacă apare dezacord cu 1 și 2
$\text{STOP} \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap \overline{1} \cap \overline{2} \cap \overline{3} \cap 4$	— sistemul se oprește
$\text{STOP} \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap \overline{1} \cap \overline{2} \cap 3 \cap \overline{4}$	— se oprește cu 3
$\text{STOP} \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap 1 \cap \overline{2} \cap \overline{3} \cap \overline{4}$	— se oprește cu 1
$\text{STOP} \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap \overline{1} \cap \overline{2} \cap \overline{3} \cap \overline{4}$	— sistemul se oprește
$\text{STOP} \cap D_1 \cap D_2 \cap \overline{D}_3 \cap \overline{D}_4 \cap \overline{1} \cap \overline{2} \cap \overline{3} \cap \overline{4}$	— sistemul este bun
$\text{STOP} \cap D_1 \cap D_2 \cap \overline{D}_3 \cap \overline{D}_4 \cap \overline{1} \cap \overline{2} \cap \overline{3} \cap \overline{4}$	— se oprește dacă apare dezacord cu 2 și 4
$\text{STOP} \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap 1 \cap \overline{2} \cap 3 \cap 4$	— se oprește dacă apare dezacord cu 1 și 4
$\text{STOP} \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap 1 \cap 2 \cap \overline{3} \cap 4$	— se oprește dacă apare dezacord cu 1, 2, și 4
$\text{STOP} \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap 1 \cap 2 \cap 3 \cap \overline{4}$	— se oprește dacă apare dezacord cu 1 și 2
$\text{STOP} \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap \overline{1} \cap \overline{2} \cap 3 \cap 4$	— se oprește dacă apare dezacord cu 4
$\text{STOP} \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap \overline{1} \cap 2 \cap \overline{3} \cap 4$	— se oprește dacă apare dezacord cu 2 și 4
$\text{STOP} \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap \overline{1} \cap 2 \cap 3 \cap \overline{4}$	— se oprește dacă apare dezacord cu 2

Tabelul 5.3 (continuare)

Starea sistemului	Comentarii
STOP $\cap$ D_1 $\cap$ D_2 $\cap$ $\overline{D}_3$ $\cap$ D_4 $\cap$ 1 $\cap$ $\overline{2}$ $\cap$ $\overline{3}$ $\cap$ 4	— se opreşte dacă apare dezacord cu 1 şi 4
STOP $\cap$ D_1 $\cap$ D_2 $\cap$ $\overline{D}_3$ $\cap$ D_4 $\cap$ 1 $\cap$ $\overline{2}$ $\cap$ 3 $\cap$ $\overline{4}$	— se opreşte dacă apare dezacord cu 1
STOP $\cap$ D_1 $\cap$ D_2 $\cap$ $\overline{D}_3$ $\cap$ D_4 $\cap$ 1 $\cap$ 2 $\cap$ $\overline{3}$ $\cap$ $\overline{4}$	— se opreşte dacă apare dezacord cu 1 şi 2
STOP $\cap$ D_1 $\cap$ D_2 $\cap$ $\overline{D}_3$ $\cap$ D_4 $\cap$ $\overline{1}$ $\cap$ $\overline{2}$ $\cap$ $\overline{3}$ $\cap$ 4	— se opreşte dacă apare dezacord cu 4
STOP $\cap$ D_1 $\cap$ D_2 $\cap$ $\overline{D}_3$ $\cap$ D_4 $\cap$ $\overline{1}$ $\cap$ $\overline{2}$ $\cap$ 3 $\cap$ $\overline{4}$	— se opreşte
STOP $\cap$ D_1 $\cap$ D_2 $\cap$ $\overline{D}_3$ $\cap$ D_4 $\cap$ 1 $\cap$ $\overline{2}$ $\cap$ $\overline{3}$ $\cap$ $\overline{4}$	— se opreşte dacă apare dezacord cu 1
STOP $\cap$ D_1 $\cap$ D_2 $\cap$ $\overline{D}_3$ $\cap$ D_4 $\cap$ $\overline{1}$ $\cap$ 2 $\cap$ $\overline{3}$ $\cap$ $\overline{4}$	— se opreşte dacă apare dezacord cu 2
STOP $\cap$ D_1 $\cap$ D_2 $\cap$ $\overline{D}_3$ $\cap$ D_4 $\cap$ $\overline{1}$ $\cap$ $\overline{2}$ $\cap$ $\overline{3}$ $\cap$ $\overline{4}$	— se opreşte
STOP $\cap$ D_1 $\cap$ $\overline{D}_2$ $\cap$ D_3 $\cap$ D_4 $\cap$ 1 $\cap$ 2 $\cap$ 3 $\cap$ 4	— se opreşte dacă apare dezacord cu 1, 3 şi 4
STOP $\cap$ D_1 $\cap$ $\overline{D}_2$ $\cap$ D_3 $\cap$ D_4 $\cap$ $\overline{1}$ $\cap$ 2 $\cap$ 3 $\cap$ 4	— se opreşte dacă apare dezacord cu 3 şi 4
STOP $\cap$ D_1 $\cap$ $\overline{D}_2$ $\cap$ D_3 $\cap$ D_4 $\cap$ 1 $\cap$ $\overline{2}$ $\cap$ 3 $\cap$ 4	— se opreşte dacă apare dezacord cu 1 şi 4
STOP $\cap$ D_1 $\cap$ $\overline{D}_2$ $\cap$ D_3 $\cap$ D_4 $\cap$ 1 $\cap$ 2 $\cap$ $\overline{3}$ $\cap$ 4	— se opreşte dacă apare dezacord cu 1 şi 4
STOP $\cap$ D_1 $\cap$ $\overline{D}_2$ $\cap$ D_3 $\cap$ D_4 $\cap$ 1 $\cap$ 2 $\cap$ 3 $\cap$ $\overline{4}$	— se opreşte dacă apare dezacord cu 1 şi 3
STOP $\cap$ D_1 $\cap$ $\overline{D}_2$ $\cap$ D_3 $\cap$ D_4 $\cap$ $\overline{1}$ $\cap$ $\overline{2}$ $\cap$ 3 $\cap$ 4	— se opreşte cu 3 şi 4
STOP $\cap$ D_1 $\cap$ $\overline{D}_2$ $\cap$ D_3 $\cap$ D_4 $\cap$ $\overline{1}$ $\cap$ 2 $\cap$ $\overline{3}$ $\cap$ 4	— se opreşte cu 4
STOP $\cap$ D_1 $\cap$ $\overline{D}_2$ $\cap$ D_3 $\cap$ D_4 $\cap$ $\overline{1}$ $\cap$ 2 $\cap$ 3 $\cap$ $\overline{4}$	— se opreşte cu 3
STOP $\cap$ D_1 $\cap$ $\overline{D}_2$ $\cap$ D_3 $\cap$ D_4 $\cap$ 1 $\cap$ $\overline{2}$ $\cap$ $\overline{3}$ $\cap$ 4	— se opreşte cu 1 şi 4
STOP $\cap$ D_1 $\cap$ $\overline{D}_2$ $\cap$ D_3 $\cap$ D_4 $\cap$ 1 $\cap$ $\overline{2}$ $\cap$ 3 $\cap$ $\overline{4}$	— se opreşte cu 1 şi 3
STOP $\cap$ D_1 $\cap$ $\overline{D}_2$ $\cap$ D_3 $\cap$ D_4 $\cap$ 1 $\cap$ 2 $\cap$ $\overline{3}$ $\cap$ $\overline{4}$	— se opreşte cu 1
STOP $\cap$ D_1 $\cap$ $\overline{D}_2$ $\cap$ D_3 $\cap$ D_4 $\cap$ $\overline{1}$ $\cap$ $\overline{2}$ $\cap$ $\overline{3}$ $\cap$ 4	— se opreşte cu 4
STOP $\cap$ D_1 $\cap$ $\overline{D}_2$ $\cap$ D_3 $\cap$ D_4 $\cap$ $\overline{1}$ $\cap$ $\overline{2}$ $\cap$ 3 $\cap$ $\overline{4}$	— se opreşte cu 3
STOP $\cap$ D_1 $\cap$ $\overline{D}_2$ $\cap$ D_3 $\cap$ D_4 $\cap$ $\overline{1}$ $\cap$ 2 $\cap$ $\overline{3}$ $\cap$ $\overline{4}$	— se opreşte
STOP $\cap$ D_1 $\cap$ $\overline{D}_2$ $\cap$ D_3 $\cap$ D_4 $\cap$ 1 $\cap$ $\overline{2}$ $\cap$ $\overline{3}$ $\cap$ $\overline{4}$	— se opreşte cu 1
STOP $\cap$ D_1 $\cap$ $\overline{D}_2$ $\cap$ D_3 $\cap$ D_4 $\cap$ $\overline{1}$ $\cap$ $\overline{2}$ $\cap$ $\overline{3}$ $\cap$ $\overline{4}$	— se opreşte
STOP $\cap$ $\overline{D}_1$ $\cap$ D_2 $\cap$ D_3 $\cap$ D_4 $\cap$ 1 $\cap$ 2 $\cap$ 3 $\cap$ 4	— se opreşte cu 2, 3 şi 4
STOP $\cap$ $\overline{D}_1$ $\cap$ D_2 $\cap$ D_3 $\cap$ D_4 $\cap$ $\overline{1}$ $\cap$ 2 $\cap$ 3 $\cap$ 4	— se opreşte cu 2, 3 şi 4
STOP $\cap$ $\overline{D}_1$ $\cap$ D_2 $\cap$ D_3 $\cap$ D_4 $\cap$ 1 $\cap$ $\overline{2}$ $\cap$ 3 $\cap$ 4	— se opreşte cu 3 şi 4
STOP $\cap$ D_1 $\cap$ D_2 $\cap$ D_3 $\cap$ D_4 $\cap$ 1 $\cap$ 2 $\cap$ $\overline{3}$ $\cap$ 4	— se opreşte cu 2 şi 4

Tabelul 5.3 (continuare)

Starea sistemului	Comentarii
$STOP \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap 1 \cap 2 \cap 3 \cap \overline{4}$	— se opreşte cu 2 şi 3
$STOP \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap \overline{1} \cap \overline{2} \cap 3 \cap 4$	— se opreşte cu 3 şi 4
$STOP \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap \overline{1} \cap 2 \cap \overline{3} \cap 4$	— se opreşte cu 2 şi 4
$STOP \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap \overline{1} \cap 2 \cap 3 \cap \overline{4}$	— se opreşte cu 2 şi 3
$STOP \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap 1 \cap \overline{2} \cap \overline{3} \cap 4$	— se opreşte cu 4
$STOP \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap 1 \cap \overline{2} \cap 3 \cap \overline{4}$	— se opreşte cu 3
$STOP \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap 1 \cap 2 \cap \overline{3} \cap \overline{4}$	— se opreşte cu 2
$STOP \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap \overline{1} \cap \overline{2} \cap \overline{3} \cap 4$	— se opreşte cu 4
$STOP \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap \overline{1} \cap 2 \cap \overline{3} \cap \overline{4}$	— se opreşte cu 3
$STOP \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap 1 \cap \overline{2} \cap \overline{3} \cap \overline{4}$	— se opreşte
$STOP \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap \overline{1} \cap \overline{2} \cap \overline{3} \cap \overline{4}$	— se opreşte
$STOP \cap D_1 \cap D_2 \cap \overline{D}_3 \cap \overline{D}_4 \cap 1 \cap 2 \cap 3 \cap 4$	— stare nesigură

Toate stările care au două elemente D defecte — oricare ar fi stările modulelor — sînt stări nesigure. Exprimîndu-se probabilitatea reuniunii stărilor nesigure, se obține:

$$1 - \mathcal{S} = (1 - R_{STOP})(1 - R_{SRH}) + R_{STOP}[R_{D_1}R_{D_2}(1 - R_{D_3})(1 - R_{D_4}) + R_{D_1}(1 - R_{D_2})R_{D_3}(1 - R_{D_4}) + R_{D_1}(1 - R_{D_2})(1 - R_{D_3})R_4 + (1 - R_{L_1})R_{D_2}R_{D_3}(1 - R_{D_4}) + (1 - R_{D_1})R_{D_2}(1 - R_{D_3}) + (1 - R_{D_1})(1 - R_{D_2})], \quad (5.52)$$

în care R_{SRH} este funcția de fiabilitate a aceluiași sistem fără semnal STOP.

Calculul funcției de fiabilitate. Analizîndu-se stările sistemului, se observă că pentru buna sa funcționare este obligatorie buna funcționare a voterului ca și a circuitelor care formează semnalul STOP.

Stările posibile ale sistemului care conduc la buna sa funcționare sînt evidențiate în tabelul 5.4. S-a utilizat același mod de notare ca în analizele precedente.

Exprimîndu-se probabilitatea reuniunii stărilor care asigură buna funcționare a sistemului, se obține expresia funcției de fiabilitate:

$$R_{SRA} = R_{STOP}R_T\{R_{D_1}R_{D_2}R_{D_3}R_{D_4}[R^4 + 4R^3(1 - R) + 6R^2(1 - R)^2] + R_{D_1}R_{D_2}R_{D_3}(1 - R_{D_4})[R^4 + 4R^3(1 - R) + 3R^2(1 - R)^2] + R_{D_1}R_{D_2}(1 - R_{D_3})R_{D_4}[R^4 + 2R^3(1 - R) + R^2(1 - R)^2] + (1 - R_{D_1})R_{D_2}R_{D_3}R_{D_4}[R^4 + 2R^3(1 - R)]\}. \quad (5.53)$$

Tabelul 5.4

Starea sistemului	Comentarii
$V \cap STOP \cap D_1 \cap D_2 \cap D_3 \cap D_4$	— 2 module din 4 trebuie să funcţioneze pentru ca sistemul să funcţioneze
$V \cap STOP \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap 1 \cap 2 \cap 3 \cap 4$	— sistem bun
$V \cap STOP \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap \overline{1} \cap 2 \cap 3 \cap 4$	— sistem bun
$V \cap STOP \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap 1 \cap \overline{2} \cap 3 \cap 4$	— sistem bun
$V \cap STOP \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap 1 \cap 2 \cap \overline{3} \cap 4$	— sistem bun
$V \cap STOP \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap 1 \cap 2 \cap 3 \cap \overline{4}$	— sistem bun
$V \cap STOP \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap \overline{1} \cap \overline{2} \cap 3 \cap 4$	— oprire posibilă
$V \cap STOP \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap \overline{1} \cap 2 \cap \overline{3} \cap 4$	— oprire posibilă
$V \cap STOP \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap \overline{1} \cap 2 \cap 3 \cap \overline{4}$	— sistem bun
$V \cap STOP \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap 1 \cap \overline{2} \cap \overline{3} \cap 4$	— se opreşte
$V \cap STOP \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap 1 \cap \overline{2} \cap 3 \cap \overline{4}$	— sistem bun
$V \cap STOP \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap 1 \cap 2 \cap \overline{3} \cap \overline{4}$	— sistem bun
$V \cap STOP \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap \overline{1} \cap \overline{2} \cap \overline{3} \cap 4$	— se opreşte
$V \cap STOP \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap \overline{1} \cap 2 \cap \overline{3} \cap 4$	— se opreşte
$V \cap STOP \cap D_1 \cap D_2 \cap D_3 \cap \overline{D}_4 \cap 1 \cap \overline{2} \cap \overline{3} \cap \overline{4}$	— se opreşte
$V \cap STOP \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap 1 \cap 2 \cap 3 \cap 4$	— sistem bun
$V \cap STOP \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap \overline{1} \cap 2 \cap 3 \cap 4$	— oprire posibilă
$V \cap STOP \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap 1 \cap \overline{2} \cap 3 \cap 4$	— oprire posibilă
$V \cap STOP \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap 1 \cap 2 \cap \overline{3} \cap 4$	— sistem bun
$V \cap STOP \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap 1 \cap 2 \cap 3 \cap \overline{4}$	— sistem bun
$V \cap STOP \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap \overline{1} \cap \overline{2} \cap 3 \cap 4$	— se opreşte
$V \cap STOP \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap \overline{1} \cap 2 \cap \overline{3} \cap 4$	— se opreşte
$V \cap STOP \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap 1 \cap \overline{2} \cap 3 \cap \overline{4}$	— se opreşte
$V \cap STOP \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap 1 \cap \overline{2} \cap \overline{3} \cap 4$	— se opreşte
$V \cap STOP \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap 1 \cap \overline{2} \cap \overline{3} \cap \overline{4}$	— se opreşte
$V \cap STOP \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap \overline{1} \cap \overline{2} \cap \overline{3} \cap 4$	— se opreşte
$V \cap STOP \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap \overline{1} \cap 2 \cap \overline{3} \cap \overline{4}$	— se opreşte
$V \cap STOP \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap 1 \cap \overline{2} \cap \overline{3} \cap \overline{4}$	— se opreşte
$V \cap STOP \cap D_1 \cap D_2 \cap \overline{D}_3 \cap D_4 \cap \overline{1} \cap \overline{2} \cap \overline{3} \cap \overline{4}$	— se opreşte
$V \cap STOP \cap D_1 \cap \overline{D}_2 \cap D_3 \cap D_4 \cap 1 \cap 2 \cap 3 \cap 4$	— se opreşte
$V \cap STOP \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap 1 \cap 2 \cap 3 \cap 4$	— sistem bun
$V \cap STOP \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap \overline{1} \cap 2 \cap 3 \cap 4$	— sistem bun
$V \cap STOP \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap 1 \cap \overline{2} \cap 3 \cap 4$	— sistem bun
$V \cap STOP \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap 1 \cap 2 \cap \overline{3} \cap 4$	— se opreşte
$V \cap STOP \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap 1 \cap 2 \cap 3 \cap \overline{4}$	— se opreşte

Tabelul 5.4 (continuare)

Starea sistemului	Comentarii
$V \cap \text{STOP} \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap \overline{1} \cap \overline{2} \cap 3 \cap 4$	— se opreşte
$V \cap \text{STOP} \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap \overline{1} \cap 2 \cap \overline{3} \cap 4$	— se opreşte
$V \cap \text{STOP} \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap \overline{1} \cap 2 \cap 3 \cap \overline{4}$	— se opreşte
$V \cap \text{STOP} \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap 1 \cap \overline{2} \cap \overline{3} \cap 4$	— se opreşte
$V \cap \text{STOP} \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap 1 \cap \overline{2} \cap 3 \cap \overline{4}$	— se opreşte
$V \cap \text{STOP} \cap \overline{D}_1 \cap D_2 \cap D_3 \cap D_4 \cap 1 \cap 2 \cap \overline{3} \cap \overline{4}$	— se opreşte
alte stări	— semnalele la ieşiri pot fi false

(b) *Structura redondantă hibridă cu două module în aşteptare*

Calculul funcţiei de securitate. Se procedează de o manieră analogă celor utilizate în cazul precedent: Rezultă:

$$1 - \mathcal{S} = (1-R_{\text{STOP}})\,(1-R_{\text{SRH}}) + R_{\text{STOP}}\,[R_{D_1} R_{D_2} R_{D_3}\,(1-R_{D_4})\,(1-R_{D_5}) + + R_{D_1} R_{D_2}\,(1-R_{D_3})\,R_{D_4}(1-R_{D_5}) + R_{D_1} R_{D_2}\,(1-R_{D_3})\,(1-R_{D_4}) + + R_{D_1}\,(1-R_{D_2})\,(1-R_{D_3}) + (1-R_{D_1})\,R_{D_2} R_{D_3} R_{D_4}\,(1-R_{D_5}) + + (1-R_{D_1})\,(1-R_{D_2}) + (1-R_{D_1})\,R_{D_2}\,(1-R_{D_3})]. \tag{5.54}$$

Calculul funcţiei de fiabilitate. Folosindu-se acelaşi procedeu ca în cazul precedent, se obţine:

$$R_{\text{SRA}} = R_{\text{STOP}}\,R_T\{R_{D_1} R_{D_2} R_{D_3} R_{D_4} R_{D_5}[R^5 + 5R^4\,(1-R) + 10R^3\,(1-R)^2 + + 10R^2\,(1-R)^3] + R_{D_1} R_{D_2} R_{D_3} R_{D_4}\,(1-R_{D_5})\,[R^5 + 5R^4\,(1-R) + + 10R^3\,(1-R)^2 + 6R^2\,(1-R)^3] + R_{D_1} R_{D_2} R_{D_3}(1-R_{D_4}) R_{D_5}[[R^5 + 5R^4(1-R) + + 7R^3\,(1-R)^2 + + 3R^2\,(1-R)^3] + R_{D_1} R_{D_2}(1-R_{D_3}) R_{D_4} R_{D_5}[R^5 + 5R^4(1-R) + + 3R^3\,(1-R)^2 + R^2(1-R)^3] + (1-R_{D_1} R_{D_2} R_{D_3} R_{D_4} R_{D_5}[R^5 + 5R^4\,(1-R) + + 7R^3\,(1-R)^2 + 3R^2\,(1-R)^3]\}. \tag{5.55}$$

5.4. MODELAREA FUNCŢIEI DE FIABILITATE CU CONSIDERAREA DEFECTĂRILOR CVASITEMPORARE

În cele ce urmează se va dezvolta un model de evaluare a probabilităţii de a avea semnale corecte la ieşirile unui circuit logic, atunci cînd se iau în considerare defectările de tip *cvasitemporar**.

* Datorită modului de operare cu semnalele logice „1" şi „0", circuitele logice evidenţiază existenţa unui defect la ieşire numai în prezenţa anumitor grupări de semnale. Acest tip de defectare este numit *defectare cvasitemporară* a circuitului considerat.

5.4.1. IPOTEZE ȘI NOTAȚII

Aplicarea metodei se bazează pe următoarele ipoteze:

- probabilitățile de defectare ale elementelor componente sînt cunoscute;
- produsele de ordin superior lui doi, inclusiv ale probabilităților de defectare ale elementelor componente se neglijează;
- defectările componentelor sînt considerate evenimente independente;
- evenimentul defectare este evidențiat în cele ce urmează prin intermediul variabilei d_k, definită astfel:

$$d_k = \begin{cases} „1“, \text{ dacă circuitul } G_k \text{ este defect,} \\ „0“, \text{ dacă circuitul } G_k \text{ funcționează corect;} \end{cases} \tag{5.56}$$

- evenimentul eroare la ieșirile E_k ale circuitului logic G_k este reprezentat prin intermediul variabilei e_k, definită cu relația:

$$e_k = \begin{cases} „1“, \text{ dacă ieșirea } E_k \text{ este eronată;} \\ „0“, \text{ dacă ieșirea } E_k \text{ nu este eronată.} \end{cases} \tag{5.57}$$

Necesitatea diferențierii celor două evenimente apare datorită faptului că pentru oricare două circuite logice, G_i și G_j, dintr-un sistem există relațiile:

$$P_r\,(d_i \cap d_j = 1) = 0,$$

conform ipotezei enunțate, iar

$P_r\,(e_i \cap e_j = 1)$ poate fi diferită de zero, deoarece defectarea unei componente se propagă la ieșirile mai multor circuite (porți logice) G din sistem.

Referitor la propagarea erorilor se fac următoarele ipoteze:

- propagarea erorii e_i la ieșirea circuitului G_k este caracterizată de funcția logică de propagare a erorii f_{ik} definită cu relația:

$$f_{ik} = \begin{cases} „1“, \text{ cînd eroarea } e_i \text{ se propagă la ieșirea } E_k, \\ „0“, \text{ cînd eroarea } e_i \text{ nu se propagă la ieșirea } E_k; \end{cases} \tag{5.58}$$

- produsul boolean al tuturor funcțiilor de propagare a unei defectări de la ieșirea E_i la ieșirea E_j,

$$F_{ij} = f_{ii_1}\, f_{i_1 i_2} \cdots f_{i_k j}, \tag{5.59}$$

poartă numele de funcție de cale și guvernează propagarea erorii e_i la ieșirea E_j.

5.4.2. CALCULUL PROBABILITĂȚII SEMNALELOR DE IEȘIRE CORECTE

În continuare se vor indica etapele algoritmului de calcul al probabilității de a avea semnale de ieșire corecte în prezența anumitor defecte în circuit.

Etapa I: *Construirea grafului propagării erorilor.* Evenimentul eroare la ieşirea E_k a unei porţi G_k corespunde reuniunii a trei evenimente mutual exclusive:

$$e_k = d_k \cup f_{ik}\, e_i \cup f_{jk}\, e_j. \tag{5.60}$$

În figura 5.15 este reprezentat graful corespunzător ecuaţiei (5.60), denumit graful propagării erorii. Nodurile corespund evenimentului eroare la intrarea şi, respectiv, ieşirea componentei G_k, iar funcţiile de propagare sînt asociate arcelor. Deoarece defectarea componentei G_k conduce la eronarea ieşirii E_k, evenimentul d_k din circuit va fi asociat nodului e_k.

Pornind de la graful de propagare a erorii pentru fiecare componentă, poate fi obţinut prin superpoziţie graful de propagare a erorilor corespunzător oricărui circuit logic complex, construind pentru fiecare variabilă de ieşire un astfel de graf. Dacă variabilele de intrare ale circuitului sînt lipsite de erori, atunci nu va exista nici un nod al grafului de propagare asociat acestora.

Etapa a II-a: *Calculul funcţiilor de cale.* Inspectînd graful de propagare a erorii se poate calcula funcţia de cale, ce guvernează propagarea defectărilor componentelor la ieşirile sistemului.

De exemplu, dacă se defectează poarta G_i (fig. 5.16*a*), atunci fiecare componentă pe calea de trecere de la G_i la G_j va avea un semnal eroare pe intrarea care face parte din calea de trecere. Cînd sistemului îi este aplicat vectorul semnal de intrare $\mathbf{X}_s$, efectul defectării d_i se va propaga la ieşirea componentei G_j dacă şi numai dacă

$$F_{ij}(\mathbf{X}_s) = 1. \tag{5.61}$$

În cazul structurii în evantai a ieşirilor unei porţi, o defectare a unei componente se propagă pe mai multe căi de circuit. Astfel, în figura 5.16*b* se arată cum o eroare la ieşirea E_i se poate propaga la ieşirea E_j pe două căi; F_{ij_1} şi F_{ij_2} sînt funcţiile de cale corespunzătoare căilor 1 şi 2. Este clar că eroarea e_i se propagă la ieşirile E_{j1} şi/sau E_{j2} dacă $F_{ij_1}(X_s)$ şi/sau $F_{ij_2}(\mathbf{X}_s)$ sînt egale cu „1" pentru cel puţin un X_s, în care caz apare un semnal eroare la ieşirea componentei G_j.

Dacă G_j este o poartă logică de tip ŞI, ŞI-NU, SAU, SAU-NU, eroarea e_i se va propaga la ieşirea E_j atît în cazul existenţei unei căi simple de propagare a erorii, cît şi a unei căi duble. Dacă G_j este o poartă logică SAU EXCLUSIV, atunci eroarea se propagă numai pe o cale simplă.

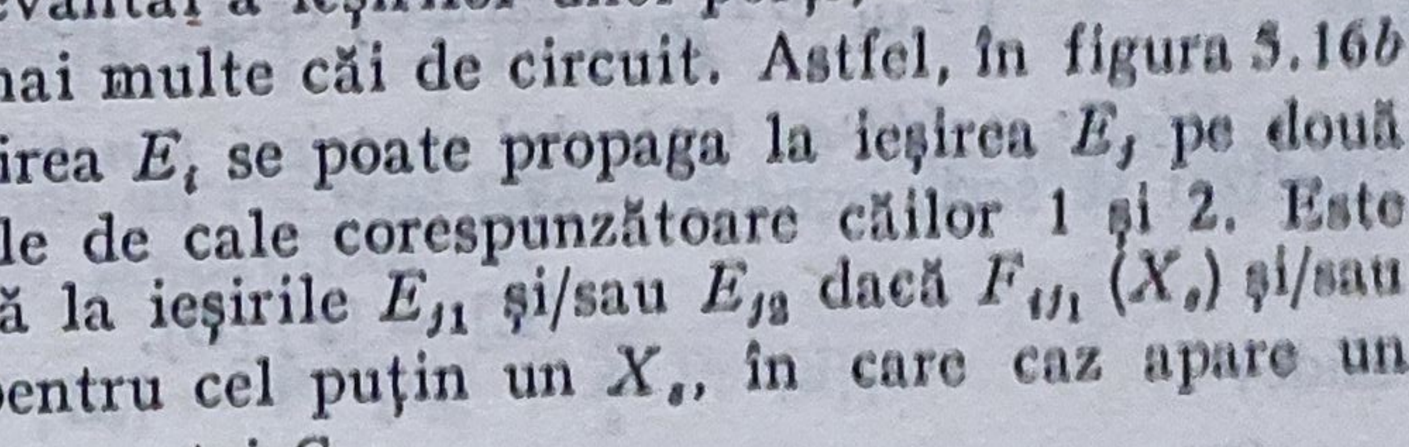


Fig. 5.15. Graful propagării erorii printr-o poartă G_k.

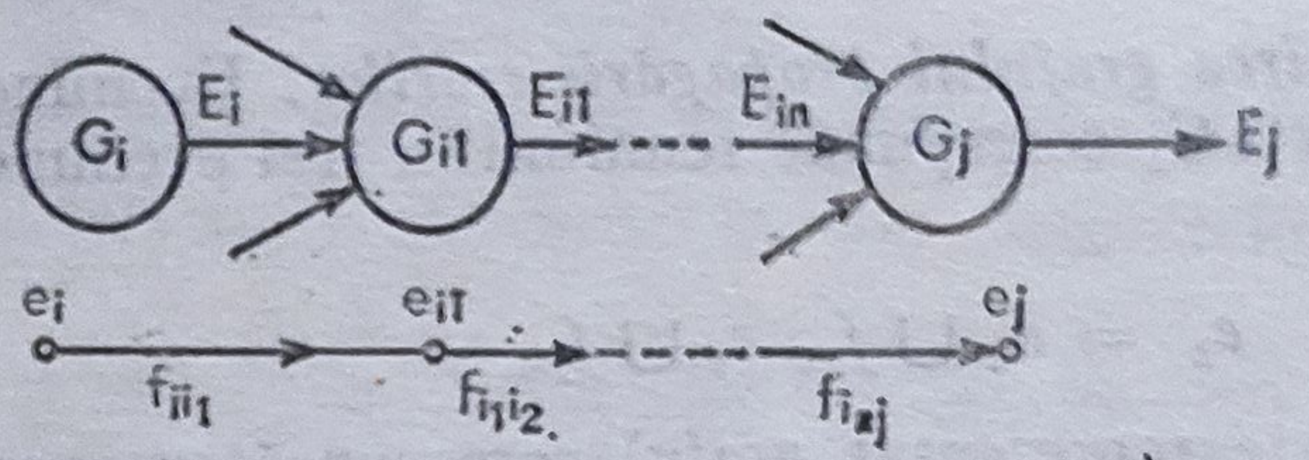


Fig. 5.16. Propagarea erorii la ieșire:
a — pe o cale; *b* — pe două căi.

Calculul funcției de cale F_{iyj} permite caracterizarea condițiilor în care defectarea d_i se propagă la ieșirea sistemului, alternînd semnalul de ieșire y_j.

De exemplu, pentru cazurile particulare indicate, pornind de la considerentele anterioare se poate determina expresia funcției de cale după cum urmează:

$$F_{ij} = \begin{cases} F_{ij1} f_{j1j} \oplus F_{ij2} f_{j1j} \oplus F_{ij1} F_{ij2}, & \text{dacă } G_i \text{ este o poartă ȘI,} \\ \text{SAU, ȘI-NU, SAU-NU;} & \\ F_{ij1} \oplus F_{ij2}, & \text{dacă } G_j \text{ este o poartă SAU EXCLUSIV.} \end{cases}$$

Etapa a III-a: *Calculul matricei funcțiilor de cale.* Pentru sistemul analizat se scrie ecuația matriceală:

$$[\mathbf{E}] = [\mathbf{F}]\,[\mathbf{D}], \tag{5.62}$$

unde:

$[\mathbf{E}] = [e_{y1}, e_{y2}, \ldots e_{ym}]^T$ este matricea erorilor la ieșirile sistemului; $[\mathbf{D}] = [d_1, \ldots, d_n]^T$ — matricea defectărilor, corespunzătoare componentelor sistemului, iar $[\mathbf{F}] = [F_{iyj}]$ — matricea funcțiilor de cale $m \times n$ dimensională.

Ecuația (5.62) permite o descriere completă a modului de propagare a defectărilor pentru orice rețea de porți logice date.

Etapa a IV-a: *Construirea tabelului defectărilor*. Se consideră că sistemul analizat are s intrări și m ieșiri. Tabelul defectărilor va avea 2^s linii corespunzătoare vectorilor semnalelor de intrare $\mathbf{X}_s$ și 2^m coloane, corespunzătoare vectorilor semnalelor răspuns, $\mathbf{Y}_m$.

Se notează cu $\mathbf{Y}^c(\mathbf{X}_s) = [y_1^c(\mathbf{X}_s), \ldots, y_m^c(\mathbf{X}_s)]$

răspunsurile obținute dacă sistemul este în stare de bună funcționare și cu $\mathbf{Y}^j(\mathbf{X}_s) = [y_1^j(\mathbf{X}_s), \ldots, y_m^j(\mathbf{X}_s)]$ răspunsurile obținute dacă în sistemul considerat poarta G_j este defectă atunci cînd sistemului se aplică la intrare vectorul semnalelor, $\mathbf{X}_s$.

Pe baza definiției date funcțiilor de cale rezultă următoarea expresie pentru determinarea variabilei $y_i^j(\mathbf{X}_s)$:

$$y_i^j(\mathbf{X}_s) = y_i^c(\mathbf{X}_s) \oplus F_{ji}(\mathbf{X}_s). \tag{5.63}$$

Se observă că atunci cînd poarta G_j este defectă, iar la intrarea sistemului se aplică $\mathbf{X}_s$, valoarea variabilei răspuns $y_i^j(X_s)$ diferă de valoarea corectă $y_i^c(\mathbf{X}_s)$ dacă $F_{ji}(\mathbf{X}_s) = 1$.

Ecuația (5.63) se poate reface pentru vectorul semnalelor de ieșire:

$$\mathbf{Y}^j(\mathbf{X}_s) = \mathbf{Y}^c(\mathbf{X}_s) \oplus F_j(\mathbf{X}_s), \tag{5.64}$$

unde $F_j(\mathbf{X}_s)$ reprezintă coloana j a matricei $\mathbf{F}$ evaluată pentru semnalele de intrare $\mathbf{X}_s$. Dispunînd de matricea $\mathbf{F}$ pentru fiecare $\mathbf{X}_s$ și calculînd fiecare element $y_i^j(\mathbf{X}_s)$, rezultă elementele (s, r) de pe linia s, coloana r a tabelului defectărilor ca avînd forma:

$$\text{elementul } (s, r) = \begin{cases} 1, \text{ dacă } \mathbf{Y}_r = \mathbf{Y}^c(\mathbf{X}_s); \\ \{d_j \mid \mathbf{Y}^j(\mathbf{X}_s) = \mathbf{Y}_r \neq \mathbf{Y}^c(\mathbf{X}_s)\}; \\ 0, \text{ în alte cazuri.} \end{cases} \tag{5.65}$$

Etapa a V-a: Calculul probabilității semnalului răspuns corect. Vectorul $\mathbf{Y}_r(\mathbf{X}_s)$ poate fi considerat ca un eveniment în spațiul $S_D = \{d_0, d_1, \ldots, d_n\}$, unde d_0 corespunde evenimentului „bună funcționare", iar $d_1, \ldots, d_n$ — evenimentului „componentă defectă". Cu această convenție $\mathbf{Y}^j(\mathbf{X}_s) = \mathbf{Y}^c(\mathbf{X}_s)$ pentru $j = 0$.

Dacă $\mathbf{Y}_r = \mathbf{Y}^j(\mathbf{X}_s)$, cînd în circuit există unul sau mai multe defecte, atunci pentru mulțimea $\{d_j\}$ se scrie:

$$\Pr\{\mathbf{Y}_r \mid \mathbf{X}_s\} = \sum_j \Pr\{d_j\}. \tag{5.66}$$

Dacă la ieșire se obține un răspuns corect în prezența defectării porții G_k, atunci acesta apare cu probabilitatea:

$$\Pr\{\mathbf{Y}^0 \mid \mathbf{X}_s\} = \Pr\{d_0\} + \sum_k \Pr\{d\}_k, \tag{5.67}$$

unde $d_k \in \{d_j \mid \mathbf{Y}^j(\mathbf{X}_s) = \mathbf{Y}^0(\mathbf{X}_s)\}$.

Dar

$$\Pr\{d_0\} = 1 - \sum_{j=1}^{n} \Pr\{d_j\}, \tag{5.68}$$

$\{d_0, d, \ldots, d_n\}$ formînd un sistem complet de evenimente.

Cu aceste observații se poate scrie:

$$\Pr\{\mathbf{Y}^0 \mid \mathbf{X}_s\} = 1 - \sum_i \Pr\{d_i\}, \tag{5.69}$$

în care $d_i \in \{d_j \mid \mathbf{Y}^j(\mathbf{X}_s) \neq \mathbf{Y}^0(\mathbf{X}_s)\}$.

Ţinîndu-se seama de rezultatele anterioare, rezultă următoarele valori ale probabilităţii de a obţine semnalele răspuns $\mathbf{Y}_r$ în prezenţa semnalelor de intrare $\mathbf{X}_s$:

$$\Pr(\mathbf{Y}_r|\mathbf{X}_s) = \begin{cases} 1 - \sum_i \Pr(d_i), \text{ unde } d_i \in \{d_j | \mathbf{Y}^j(\mathbf{X}_s) \neq \mathbf{Y}^c(\mathbf{X}_s)\}, \\ \text{dacă elementul } (s, r) = 1; \\ \sum_j \Pr(d_j), \text{ dacă elementul } (s, r) = \sum_j d_j; \\ 0, \text{ dacă elementul } (s, r) = 0 \end{cases} \tag{5.70}$$

Este deci posibil să se obţină valoarea funcţiei de fiabilitate a sistemului (circuit logic), avîndu-se în vedere tipurile de defecţări cvasitemporare specifice acestora:

$$Rx_s = 1 - \sum_i \Pr(d_i), \tag{5.71}$$

d_i fiind defectările care conduc la răspunsuri eronate cînd se aplică semnalele $\mathbf{X}_s$.

Exemplul 5.3. Se consideră circuitul secvenţial din figura 5.17*a*. Se construieşte graful de propagare a erorilor (fig. 5.17*b*). Pe această bază se pot scrie imediat ecuaţiile:

$$e_{y_1} = e_1 = d_1;$$

$$e_{y_2} = e_M \cup e_4 = e_M \cup d_4 \cup E_2 e_3 \cup E_3 e_2 =$$

$$= d_M \cup d_4 \cup E_2 (e_1 \cup d_3) \cup E_3 d_2 = d_M \cup d_4 \cup E_2 (d_1 \cup d_3) \cup E_3 d_2;$$

$$E_2 = x_1 + x_2;$$

$$E_3 = \overline{x_1 x_2},$$

ecuaţii care pot fi puse sub forma matriceală (5.62):

$$\begin{bmatrix} e_{y_1} \\ e_{y_2} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ E_2 & E_3 & E_2 & 1 & 1 \end{bmatrix} [d_1 \; d_2 \; d_3 \; d_4 \; d_M]^T.$$

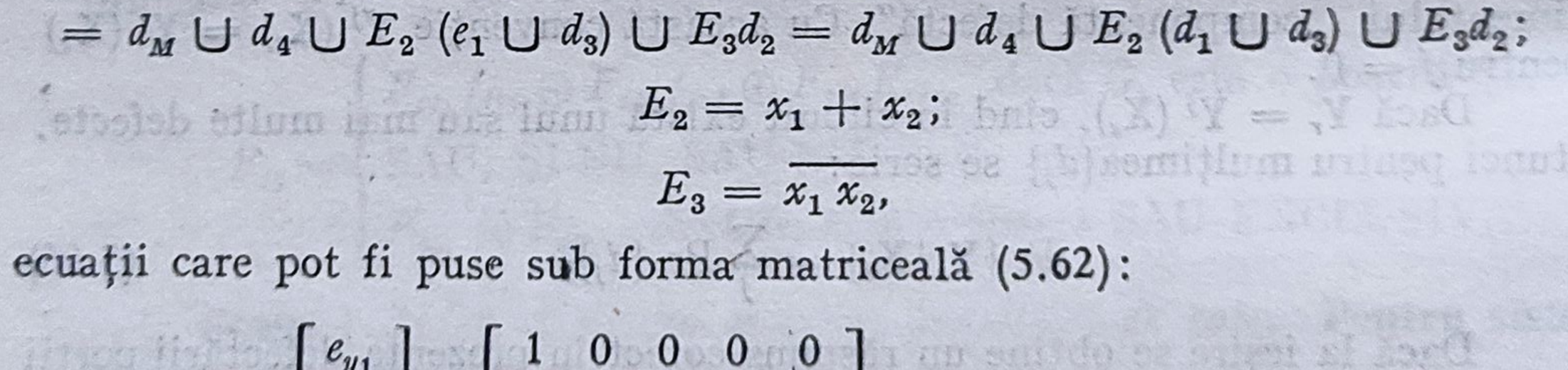


Fig. 5.17. Construirea grafului propagării erorii pentru un circuit:
a — circuitul; *b* — graful propagării erorilor.

Se construiește tabelul defectărilor (tabelul 5.5). Elementele sale se obțin imediat pe baza ecuațiilor 5.64 și 5.65.

Se cunoaște Pr $(d_j) = a$, pentru $j = 1, 2, 3, 4$, și Pr $(d_M) = b$. Utilizîndu-se ecuația 5.70, rezultă probabilitatea de a avea un set de semnale de ieșire atunci cînd la intrarea sistemului se aplică setul de semnale $\mathbf{X}_s$. Rezultatele sînt concentrate în tabelul 5.6. De aici, se pot identifica ușor valorile funcțiilor de fiabilitate condiționate de diferite combinații ale semnalelor de intrare. Astfel, atunci cînd:

— la intrare se aplică semnalele 0, 0:

$$R_{0,0} = 1 - 3a - b;$$

— la intrare se aplică semnalele 0,1:

$$R_{0,1} = 1 - 4a - b;$$

— la intrare se aplică semnalele 1, 0:

$$R_{1,0} = 1 - 4a - b;$$

— la intrare se aplică semnalele 1, 1:

$$R_{1,1} = 1 - 3a - b.$$

Tabelul 5.5

$\mathbf{X}s = (x_1, x_2)$	$\mathbf{Y}r = (y_1, y_2)$			
	0 0	0 1	1 0	1 1
0 0	1	d_2, d_4, d_M	d_1	0
0 1	d_2, d_3, d_4, d_M	1	d_1	0
1 0	d_2, d_3, d_4, d_M	1	d_1	0
1 1	0	d_1	1	d_3, d_4, d_M

Tabelul 5.6

$\mathbf{X}s = (x_1, x_2)$	$\mathbf{Y}r = (y_1, y_2)$			
	0 0	0 1	1 0	1 1
0 0	$1-3a-b$	$2a+b$	a	0
0 1	$3a+b$	$1-4a-b$	a	0
1 0	$3a+b$	$1-4a-b$	a	0
1 1	0	a	$1-3a-b$	$2a+b$

5.5. EVALUAREA PERFORMANȚELOR DE FIABILITATE PENTRU SISTEMELE TOLERANTE LA DEFECTĂRI REPARABILE. STUDII DE CAZ

În cele ce urmează vor fi investigate performanțele de fiabilitate ale unor calculatoare tolerante la defectări — sisteme reparabile —, utilizîndu-se ca bază de comparație indicatorii coeficientul de disponibilitate și media timpului de funcționare între defectări. Modelele folosite pentru analiza de fiabilitate se bazează pe procesele de reînnoire de tip Markov.

5.5.1. IPOTEZE ȘI NOTAȚII

Se fac următoarele ipoteze comune modelelor dezvoltate în continuare:

(1) Fiecare sistem este alcătuit din patru unități, A_i $(i = 1, 2)$ și B_i $(i = 1, 2)$. În exemplele următoare A_1 și A_2 sînt unități de procesare, identice, iar B_1 și B_2 sînt unități de memorie, identice;

(2) Unitățile A_i $(i = 1, 2)$ și B_i $(i = 1, 2)$ au o repartiție exponențială a timpului de bună funcționare cu ratele de defectare — constante în timp — λ_1 și, respectiv, λ_2, atunci cînd acestea funcționează, și $a\lambda_1$, respectiv $a\lambda_2$ $(0 \leqslant a \leqslant 1)$, atunci cînd acestea sînt în așteptare pentru a fi comutate. Unitățile A_i $(i = 1, 2)$ și B_i $(i = 1, 2)$ au funcțiile de repartiție a timpului de reparare arbitrare, respectiv $G_1(t)$ și $G_2(t)$ cu valorile medii $\frac{1}{\gamma_1}$ și, respectiv $\frac{1}{\gamma_2}$;

(3) Facilitatea reparațiilor este: un singur reparator cu disciplina serviciului; „primul venit, primul servit";

(4) Orice comutare a unei rezerve este perfectă și instantanee. Se introduc următoarele notații.

$\mathbf{Z}_i$ — vectorul stărilor pentru modelul i;
$\mathbf{Rg}_i$ — mulțimea punctelor de regenerare pentru modelul i;
$F_i^*(s)$ — transformarea Laplace-Stieltjes pentru funcția $F_i(t)$;
$\mathbf{NRg}_i$ — mulțimea punctelor de neregenerare pentru modelul i;
$\mathbf{A}_i$ — vectorul stărilor de operare pentru modelul i;
$\mathbf{Q}_i$ — matricea probabilității de tranziție pentru modelul i;
A_i — coeficientul de disponibilitate pentru modelul i;
M_i — media timpului de funcționare între defectări (MTBF) pentru modelul i;
$\mathcal{F}_i$ — numărul de defectări așteptat pe unitatea de timp pentru sistemul i;

$$0_0 = 2\lambda_1 + \lambda_2;$$

$$0_1 = \lambda_1 + 2\lambda_2;$$

$$0_2 = 2\lambda_1 + \lambda_2;$$

$$\theta_a = \lambda_1 + \lambda_2 + a(\lambda_1 + \lambda_2);$$

$$\theta_s = 2\lambda_1 + 2\lambda_2 + \sigma_1 + \sigma_2;$$

$$[\cdot]'|_{s=0} = \lim_{s \to 0} d\,[\cdot]/ds.$$

5.5.2. DETECȚIA DEFECTĂRILOR ȘI COMUTAREA REZERVELOR SÎNT CONSIDERATE PERFECTE

5.5.2.1. DIAGRAMA TRANZIȚIILOR ÎNTRE STĂRI

M o d e l u l 1. *Sistemul neredondant.* Se consideră sistemul neredondant reprezentat în figura 5.18*a*. Acest sistem este studiat spre a servi drept criteriu de comparație pentru analizele fiabilistice ulterioare ale structurilor tolerante la defectări. Evident, sistemul neredondant considerat se defectează atunci cînd se defectează una dintre unitățile sale, A_1 sau B_1.

În figura 5.18*b* se prezintă diagrama tranzițiilor între stările acestui sistem. Pot fi evidențiate:

— starea 0: sistemul este operant;

— starea 1: unitatea A_1 este defectă (sistemul este defect);

— starea 2: unitatea B_1 este defectă (sistemul este defect).

M o d e l u l 2. *Sistem cu structură redondantă de comutație* (*sistem duplex*). Structura acestui sistem este indicată în figura 5.19*a*, iar diagrama tranzițiilor posibile între stările sistemului considerat, în figura 5.20. Pot fi evidențiate următoarele stări ale sistemului studiat:

— starea 0: ambele sisteme sînt operante;

— starea 1: A_i ($i = 1$ sau 2) se defectează, unitatea defectă este reparată imediat — sistemul este în stare de bună funcționare, fiind însă funcțional un singur calculator;

— starea 2: B_i ($i = 1$ sau 2) se defectează, unitatea defectată se repară imediat — sistemul este în stare de funcționare, fiind funcțional numai un calculator;

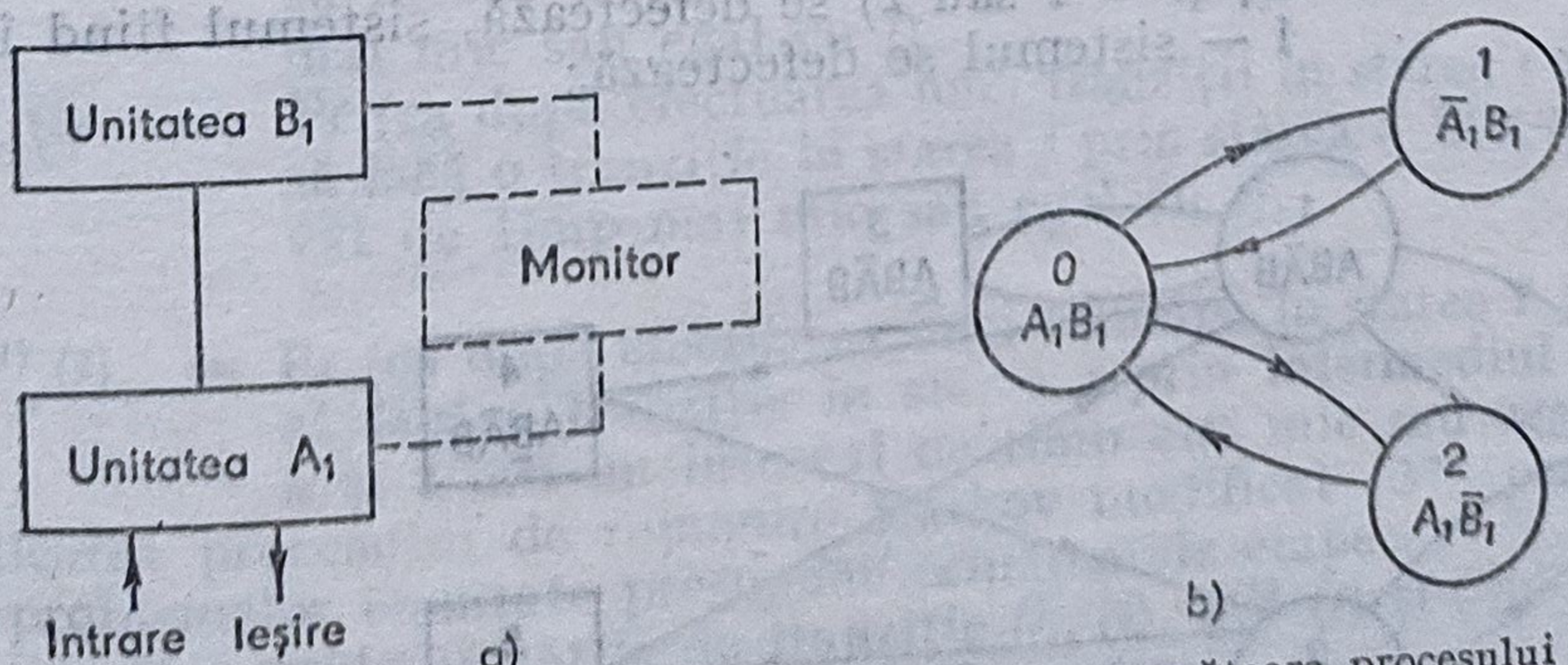


Fig. 5.18. Construirea diagramei stărilor corespunzătoare procesului Markov analizat:

a — configurația sistemului; *b* — diagrama tranziției între stări.

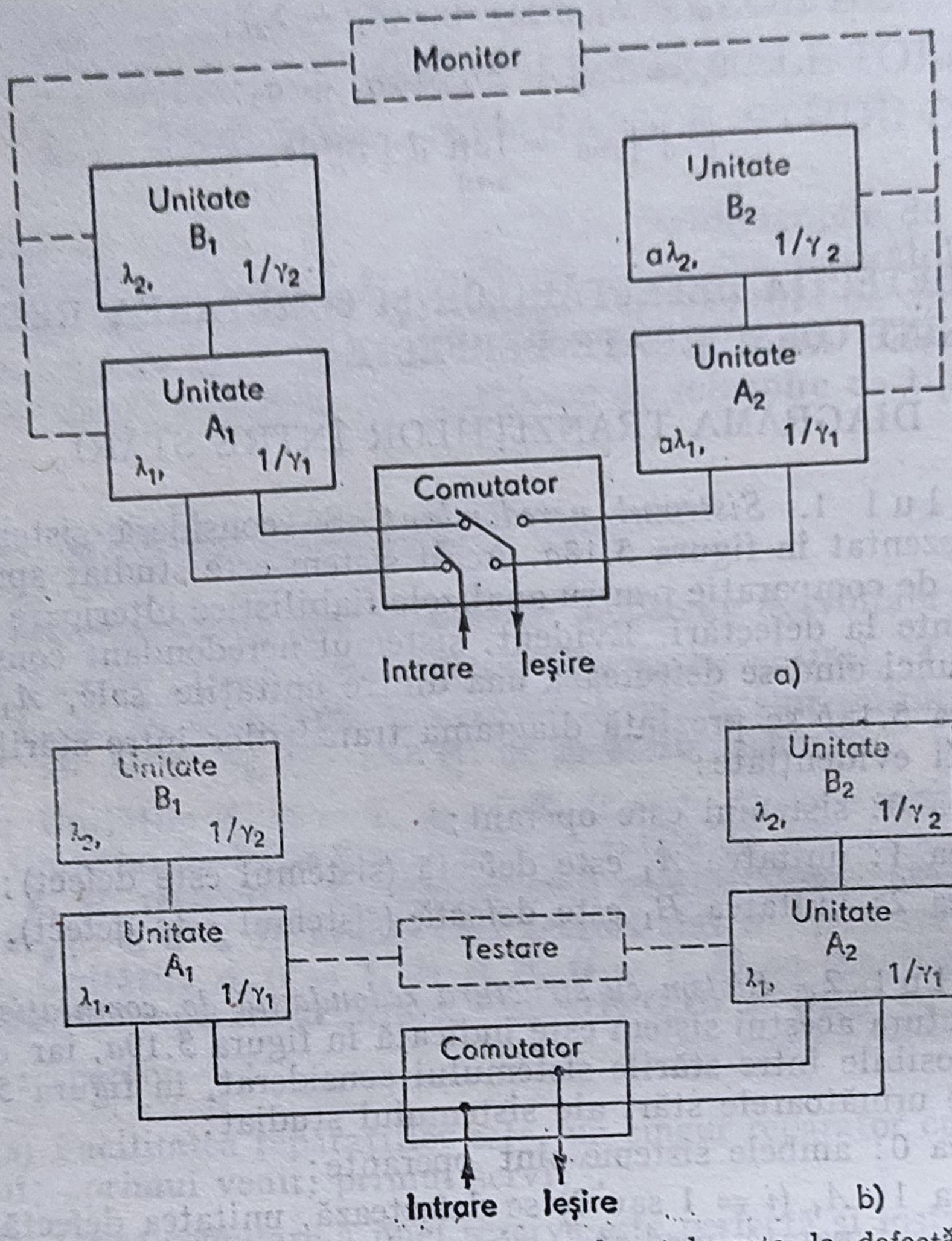


Fig. 5.19. Configurația calculatoarelor tolerante la defectări analizate:
a — sistemul duplex; *b* — sistemul dual.

— starea 3: A_i ($i = 1$ sau 2) se defectează, sistemul fiind în starea 1 — sistemul se defectează;
— starea 4: B_i ($i = 1$ sau 2) se defectează, sistemul fiind în starea 1 — sistemul se defectează;

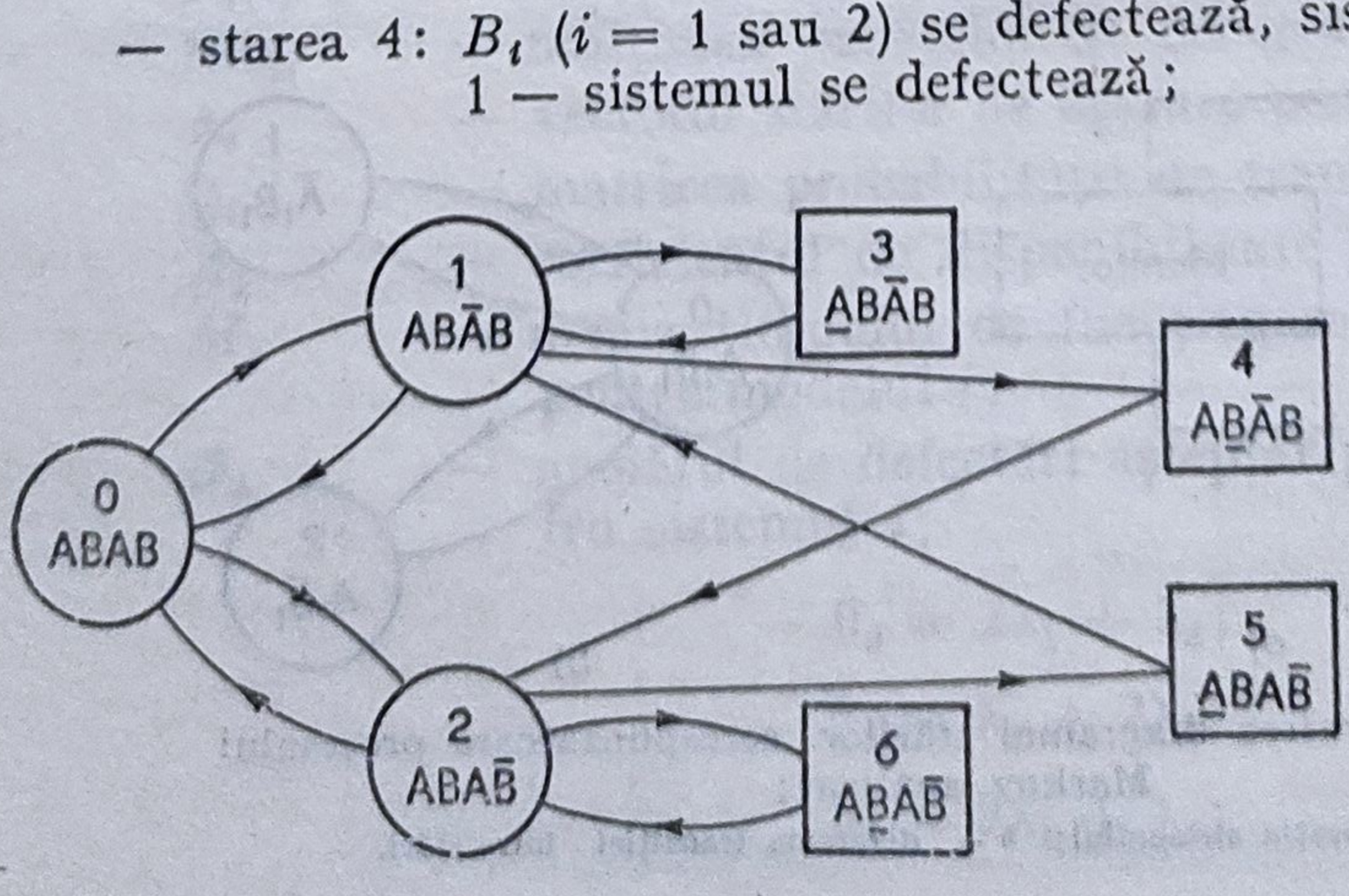


Fig. 5.20. Diagrama tranziției între stări pentru sistemul duplex analizat.

— starea 5: A_i ($i = 1$ sau 2) se defectează, sistemul fiind în starea 2 — sistemul se defectează;

— starea 6: B_i ($i = 1$ sau 2) se defectează, sistemul fiind în starea 2 — sistemul se defectează.

Stările $i = \{0, 1, 2\}$ sînt puncte de regenerare, iar stările $i = \{3, 4, 5, 6\}$ — puncte de neregenerare în tranzițiile stărilor. În diagrama din figura 5.20 s-au reprezentat cu simbolurile „‾" și „_" evenimentele „unitatea defectă este în reparație", respectiv „unitatea defectă așteaptă pentru a fi reparată".

M o d e l u l 3. *Sistem cu structură redondantă generală* (*sistem dual*). Schema sistemului considerat este indicată în figura 5.19*b*. Așa cum s-a arătat în capitolul 2, ambele calculatoare funcționează *on line*, iar defectarea unei unități A_i sau B_i este mascată instantaneu la ieșire.

Stările sistemului sînt aceleași ca mai înainte, numai că probabilitățile de trecere dintr-o stare în alta pot diferi.

5.5.2.2. CALCULUL FUNCȚIEI DE DISPONIBILITATE ȘI AL MEDIEI TIMPULUI DE FUNCȚIONARE ÎNTRE DEFECTĂRI

Dacă toate stările sistemelor descrise anterior constituie puncte de regenerare este posibil să se utilizeze modelarea convențională bazată pe procesele de reînnoire de tip Markov [14].

Totuși, deoarece pentru unele dintre modelele analizate se presupun repartiții arbitrare ale timpilor de reparare (diferite de cea uzuală — exponențială) intervin și stări care nu sînt puncte de regenerare. Pentru a soluționa această problemă se va utiliza o modelare a proceselor de reînnoire bazate pe procesele Markov, modificată în conformitate cu [37].

Astfel, dacă se consideră tranziția de la i la j, starea de pornire i fiind un punct de regenerare, se poate scrie probabilitatea tranziției într-o etapă, $Q_{ij}(t)$. Dacă stările de pornire k și l nu sînt puncte de regenerare se introduce probabilitatea de tranziție în doi pași $Q_{ij}^{(k)}(t)$ sau probabilitatea de tranziție în trei pași $Q_{ij}^{(k,l)}(t)$; starea i trebuie să fie un punct de regenerare. Cele trei probabilități de tranziție se definesc cu următoarele relații:

$Q_{ij}(t)$ = Pr (ca după efectuarea unei tranziții în starea i procesul să facă o tranziție în starea j într-un interval de timp mai mic sau egal cu t),

$Q_{ij}^{(k)}(t)$ = Pr (ca după efectuarea unei tranziții în starea i procesul să facă o tranziție în starea j prin starea k într-un interval de timp mai mic sau egal cu t),

și

$Q_{ij}^{(k,l)}(t)$ = Pr (ca după efectuarea unei tranziții în starea i procesul să facă o tranziție în starea j prin intermediul stărilor k și l într-un interval de timp mai mic sau egal cu t).

Aplicarea procesului de reînnoire Markov modificat [37] pentru rezolvarea problemelor enunțate presupune următoarele etape:

(1) Se obțin probabilitățile de tranziție $Q_{ij}(t)$, $Q_{ij}^{(k)}(t)$ și $Q_{ij}^{(k,l)}(t)$ pentru toate stările considerate;

(2) Se construiesc transformatele Laplace ale acestor probabilități; fie acestea $Q_{ij}^{*}(s)$, $Q_{ij}^{*(k)}(s)$ și $Q_{ij}^{*(k,l)}(s)$;

(3) Se consideră probabilităţile de tranziţie limită ca fiind:

$$Q_{ij} = \lim_{s \to 0} \; Q_{ij}^{*}(s), \tag{5.72}$$

$$Q_{ij}^{(k)} = \lim_{s \to 0} \; Q_{ij}^{*(k)}(s), \tag{5.73}$$

$$Q_{ij}^{(k,l)} = \lim_{s \to 0} \; Q_{ij}^{*(k,l)}(s) \tag{5.74}$$

şi se caută probabilităţile limită $\pi_i (i = 0, 1, \ldots, m)$, unde π_i indică probabilitatea de a fi în starea i după un număr infinit de tranziţii [14] pentru lanţurile Markov construite:

$$\sum_{k=D}^{m} \pi_k = 1, \qquad \pi = \pi\,[Q], \tag{5.75}$$

în care $\pi = (\pi_0, \pi_1, \ldots \pi_m)$ este vectorul probabilităţilor limită, iar Q — matricea probabilităţilor de tranziţie compusă din punctele de regenerare posibile, $i = 0,1,\ldots,m$, adică se neglijează punctele de neregenerare:

$$[Q] = \begin{bmatrix} \tilde{Q}_{00} & \tilde{Q}_{01} & \ldots & \tilde{Q}_{0} \\ \tilde{Q}_{10} & \tilde{Q}_{11} & \ldots & \tilde{Q}_{1} \\ \tilde{Q}_{m0} & \tilde{Q}_{m1} & \ldots & \tilde{Q}_{mm} \end{bmatrix}, \tag{5.76}$$

unde $\tilde{Q}_{ij}$ poate fi obţinut prin neglijarea punctelor de neregenerare;

(4) Se obţin mediile necondiţionale $\mu_i (i = 0,1,\ldots,m)$, care indică media timpului de rămînere a procesului în starea i [14], neglijînd în continuare punctele de neregenerare;

(5) Se calculează media timpului de reîntoarcere din starea i în starea i cu relaţia:

$$l_{ii} = \sum_{k=0}^{m} \pi_k \mu_k / \pi_i \; (i = 0,1,\ldots m); \tag{5.77}$$

(6) Se obţin mediile necondiţionale de neregenerare, l_i, $(i = 0,1,\ldots, \ldots,m)$ luîndu-se în considerare şi punctele de neregenerare;

(7) Se calculează probabilităţile limită p_i, utilizîndu-se media timpilor de reîntoarcere şi mediile necondiţionale;

(8) Coeficientul de disponibilitate, A_i (pentru modelul i), poate fi obţinut prin trecerea în revistă a probabilităţilor limită pentru stările de operare ale sistemului;

(9) Se obţine $\mathcal{T}_i$ — pentru modelul i — care reprezintă numerele aşteptate de defectări ale sistemului în unitatea de timp în starea sigură, folosindu-se media timpului de reîntoarcere şi probabilităţile limită;

(10) Media timpului între defectări, M_i, se obţine cu relaţia:

$$M_i = A_i / \mathcal{T}_i. \tag{5.77}$$

Modelul 1. *Sistemul neredondant*. Deoarece stările definite în § 5.5.2.1 pentru acest sistem sînt puncte de regenerare, se vor utiliza procesele de reînnoire de tip Markov convenţionale.

Avîndu-se în vedere repartiţiile adoptate pentru timpii de bună funcţionare şi reparare se obţin probabilităţile tranziţiilor $O_{ij}(t)$ $(i,j \in \mathbf{Z}_1)$; se determină transformatele Laplace, care în cazul de faţă sînt:

$$Q_{0i}^{*}(s) = \lambda_i / (s + \theta_0); \tag{5.78}$$

$$Q_{i0}^{*}(s) = G_i(s), \; i = 1,2.$$

Pe baza rezultatelor anterioare rezultă:

$$A_1 = \mu_0 / l_{00} = (\lambda_1/\gamma_1 + \lambda_2/\gamma_2 + 1)^{-1}, \tag{5.79}$$

unde μ_0 este dat de:

$$[1 - Q^*_{01}(s) - Q^*_{02}(s)]' \Big|_{s=0} = \frac{1}{\lambda_1 + \lambda_2}, \tag{5.80}$$

iar l_{00} este timpul mediu de revenire în starea 0 dat de:

$$l_{00} = [1 - Q^*_{01}(s)\, Q^*_{10}(s) - Q^*_{02}(s)\, Q^*_{20}(s)]' \Big|_{s=o} = \frac{1}{\theta_0}\left(\frac{\lambda_1}{\gamma_1} + \frac{\lambda_2}{\gamma_2} + 1\right). \tag{5.81}$$

Pornind de la rezultatele anterioare — relația (5.77) — se obține:

$$M_1 = \frac{1}{\theta_0}.$$

M o d e l e l e 2 ș i 3 . (*Sistemele duplex și dual*). Cele două modele pot fi analizate în mod similar, ele prezentînd diferențe doar în ceea ce privește valorile ratelor de defectare. În cazul sistemului duplex ratele de defectare pentru elementele componente sînt $a\lambda_1$ și, respectiv $a\lambda_2$, pe cînd în ceea ce privește sistemul dual ratele de defectare sînt λ_1 și, respectiv, λ_2, $a = 1$, elementele funcționînd în paralel.

Mai întîi se scriu expresiile probabilităților de tranziție $Q_{ij}(t)$ ($i \in \in \mathbf{Rg}_2$, $j \in \mathbf{Z}_2$) și $Q^{(k)}_{ij}(t)$ ($i \in \mathbf{Rg}_2$, $k \in \mathbf{NRg}_2$, $j \in \mathbf{Z}_2$), după care se obțin transformatele Laplace ale acestora.

$$Q^*_{0i}(s) = \frac{(1+a)\lambda_i}{s + \theta_0}, \ (i = 1,2); \tag{5.82}$$

$$Q^*_{i0}(s) = G^*_i(s + \theta_0), \ (i = 1,2); \tag{5.83}$$

$$Q^*_{i,j+2i}(s) = \frac{\lambda_j}{s + \theta_0}\,[1 - G^*_i(s + \theta_0)]\ (i,j = 1,2); \tag{5.84}$$

$$Q^{*(j+2i)}_{ij}(s) = \frac{\lambda_j}{s + \theta_0}\,[G^*_i(s) - G^*_i(s + \theta_0)], \ (i,j = 1,2). \tag{5.85}$$

Pe baza rezultatelor anterioare și ținîndu-se seama că stările 0,1 și 2 sînt puncte de regenerare, se poate scrie următoarea matrice a probabilităților de tranziție corespunzătoare lanțului Markov:

$$[\mathbf{Q}_2 =] \begin{vmatrix} 0 & Q_{01} & Q_{02} \\ & {\scriptstyle (3)} & {\scriptstyle (4)} \\ Q_{10} & Q_{11} & Q_{12} \\ & {\scriptstyle (6)} & {\scriptstyle (5)} \\ Q_{20} & Q_{22} & Q_{21} \end{vmatrix} \tag{5.86}$$

unde

$$Q_{0i} = \frac{(1+a)\lambda_i}{0_a}, \ (i = 1,2); \tag{5.87}$$

$$Q_{i0} = G^*_i(0_0), \ (i = 1,2); \tag{5.88}$$

$$Q_{ij\,(j+2i)} = \frac{\lambda_j}{0_0}\,[1 - G^*_i(0_0)], \ (i,j = 1,2). \tag{5.89}$$

Rezolvînd ecuațiile

$$[\pi_0, \pi_1, \pi_2] = [\pi_0, \pi_1, \pi_2]\,[Q_2] \tag{5.90}$$

și

$$\sum_{k \in Rg_2} \pi_k = 1.$$

se obțin probabilitățile limită π_i $(i \in \mathbf{Rg}_2)$ pentru lanțul Markov analizat.

Mediile μ_i, $(i \in \mathbf{Rg}_2)$ — neglijîndu-se de neregenerare — sînt date de ecuațiile:

$$\mu_0 = [1 - Q_{01}^*(s) - Q_{02}^*(s)]'_{s=0} = \frac{1}{\Theta_a}; \tag{5.91}$$

$$\mu_i = [1 - Q_{i0}^*(s) - Q_{i,1}^{*(1+2i)}(s) - Q_{i,2}^{*(2+2i)}(s)]_{s=0} = \frac{1}{\Theta_i} \tag{5.92}$$

În continuare se calculează mediile timpilor de revenire:

$$l_{ii} = \sum_{k \in Rg_2} \frac{\pi_k \mu_k}{\pi_i}, \quad (i \in \mathbf{Rg}_2). \tag{5.93}$$

Mediile necondiționale ζ_i, $(i \in \mathbf{A}_2)$ — cînd nu se mai neglijează punctele de neregenerare — se obțin din ecuațiile:

$$\zeta_0 = \mu_0 = \frac{1}{\theta_a}; \tag{5.94}$$

$$\zeta_i = [1 - Q_{i0}^*(s) - Q_{i,1+2i}^*(s) - Q_{i,2+2i}^*(s)]'_{s-0} =$$

$$= \frac{1}{\theta_0}[1 - G_i^*(\theta_0)], \quad (i = 1,2). \tag{5.95}$$

Pornind de la rezultatele anterioare se calculează probabilitățile limită pentru stările de operare, astfel:

$$P_i = \frac{\zeta_i}{l_{ii}}, \quad (i \in \mathbf{A}_2) \tag{5.96}$$

De aici rezultă coeficientul de disponibilitate:

$$A_2 = \sum_{k \in A_2} P_k. \tag{5.97}$$

Mai departe se calculează numărul așteptat de defectări ale sistemului pe unitatea de timp în starea sigură:

$$\mathcal{F}_2 = \frac{\sum_{k=1}^{2} (Q_{k,1+2k} + Q_{k,2+2k})}{l_{kk}}. \tag{5.98}$$

Pornindu-se de la aceste rezultate, se obține în continuare $M_2(\mathrm{MTBF}_2)$ corespunzător sistemului analizat:

$$\mathrm{MTBF}_2 = \frac{A_2}{\mathcal{F}_2}. \tag{5.99}$$

E x e m p l u l 5.4. În cele ce urmează se vor ilustra cantitativ analizele întreprinse pentru cele trei module studiate. Se consideră că timpii de reparare pentru elementele componente sînt repartizați după o lege gamma cu parametrul de formă egal cu 2, atît pentru unitățile A cît și pentru cele B:

$$G_i(t) = 1 - (1 + 2\gamma_i t)\exp(-2\gamma_i t),\ (i = 1,2).$$

Rezultatele analizei sînt sistematizate în graficele din figurile 5.21 ÷ ÷ 5.23.

Astfel, în figura 5.21 este indicată dependența indisponibilității fiecărui sistem de media timpului de reparare $\frac{1}{\gamma_2}$, corespunzătoare unității B_i $(i = 1,2)$, în cazul în care timpii medii de funcționare pentru unitățile A_i, $\frac{1}{\lambda_1}$, și B_i, $\frac{1}{\lambda_2}$, sînt 800 și 1 000 ore, respectiv 1 600 și 2 000 ore. Indisponibilitatea este definită de relația $1 - A_i$ $(i = 1, 2, 3)$, A_i fiind coeficientul de disponibilitate pentru sistemul i. Se constată ușor că cele două sisteme cu structura redondantă (curbele *2* și *3*) au o disponibilitate mai ridicată decît sistemul neredondant (curba *1*). De asemenea se observă imediat că din punctul de vedere al disponibilității sistemul dual $(a = 1)$ prezintă performanțe inferioare sistemului duplex (cazul $a = 0$). În figura 5,22 se prezintă dependența coeficientului de disponibilitate al sistemului

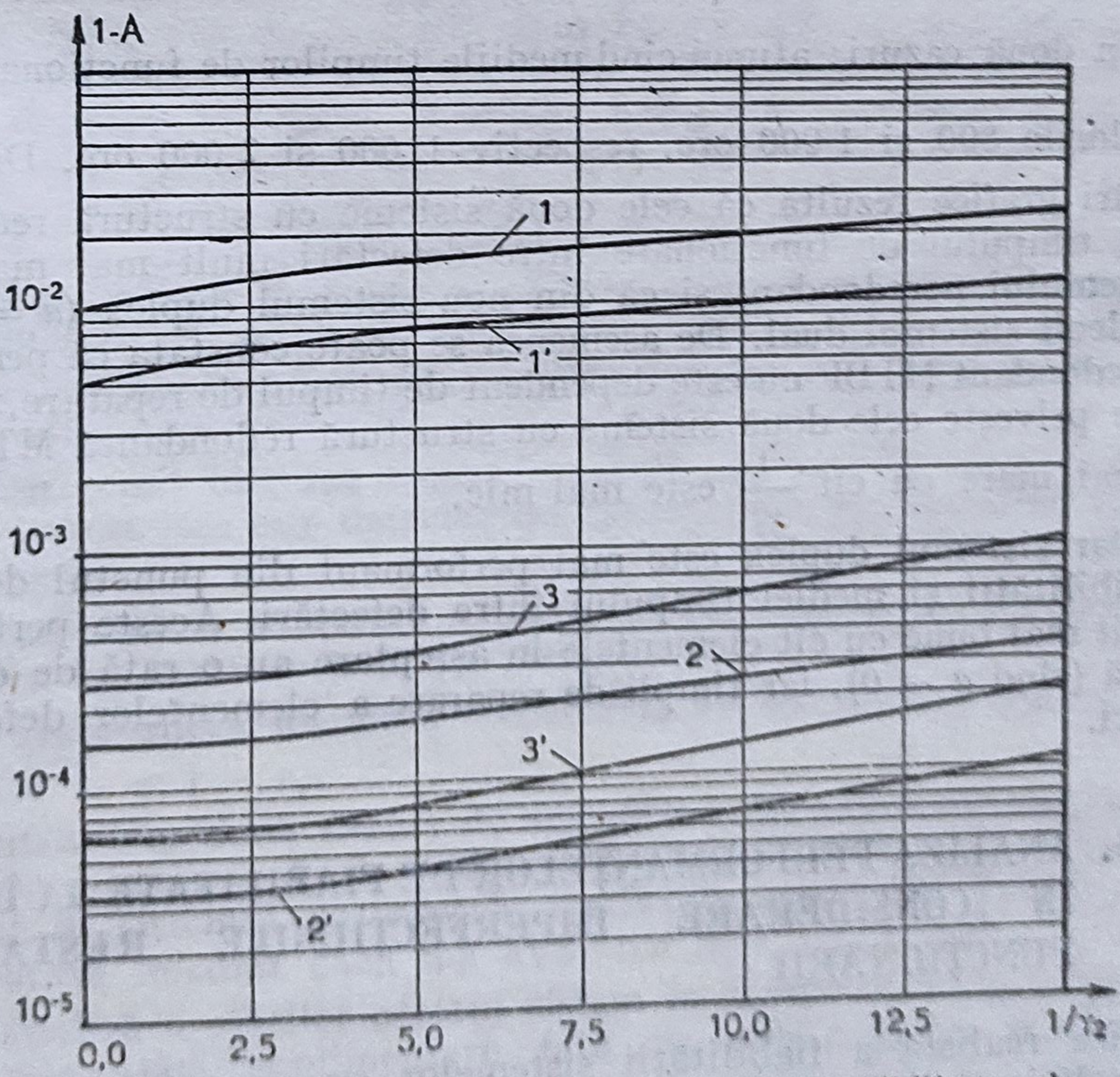


Fig. 5.21. Variația coeficientului de indisponibilitate al sistemelor analizate cu variația mediei timpului de reparare.

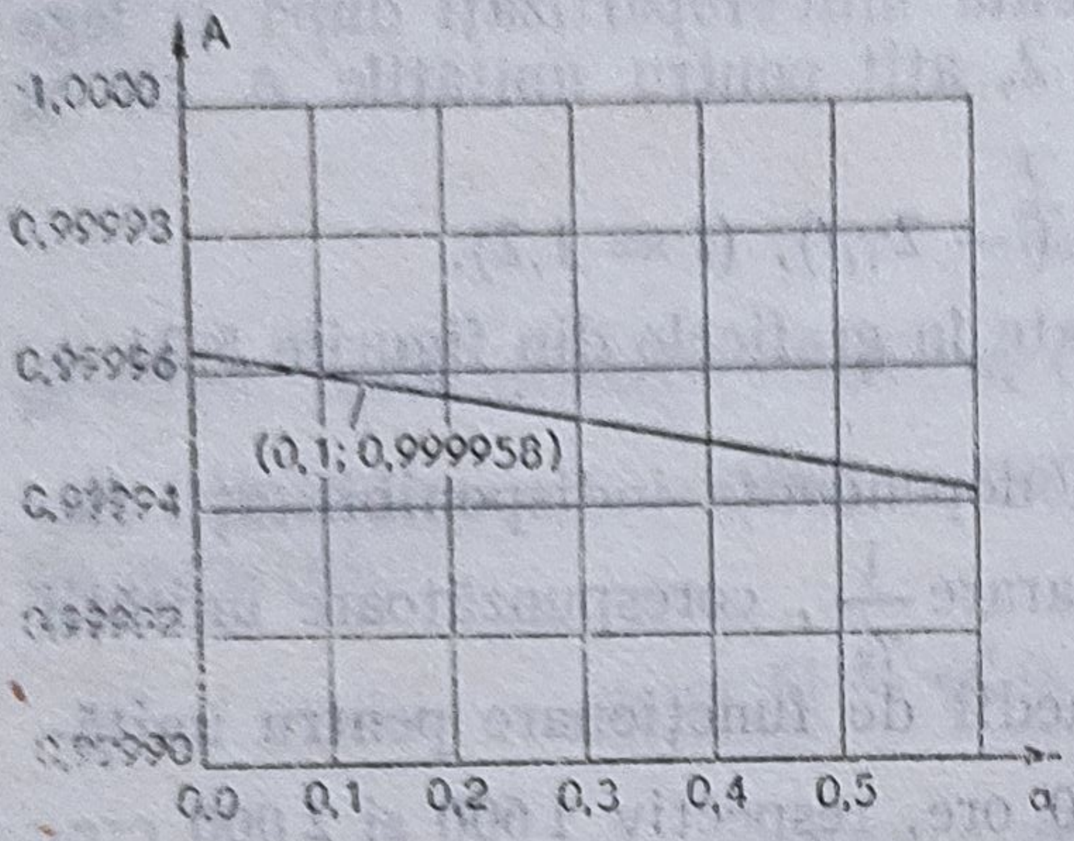


Fig. 5.22. Dependența coeficientului de disponibilitate al sistemului duplex analizat de valoarea coeficientului a.

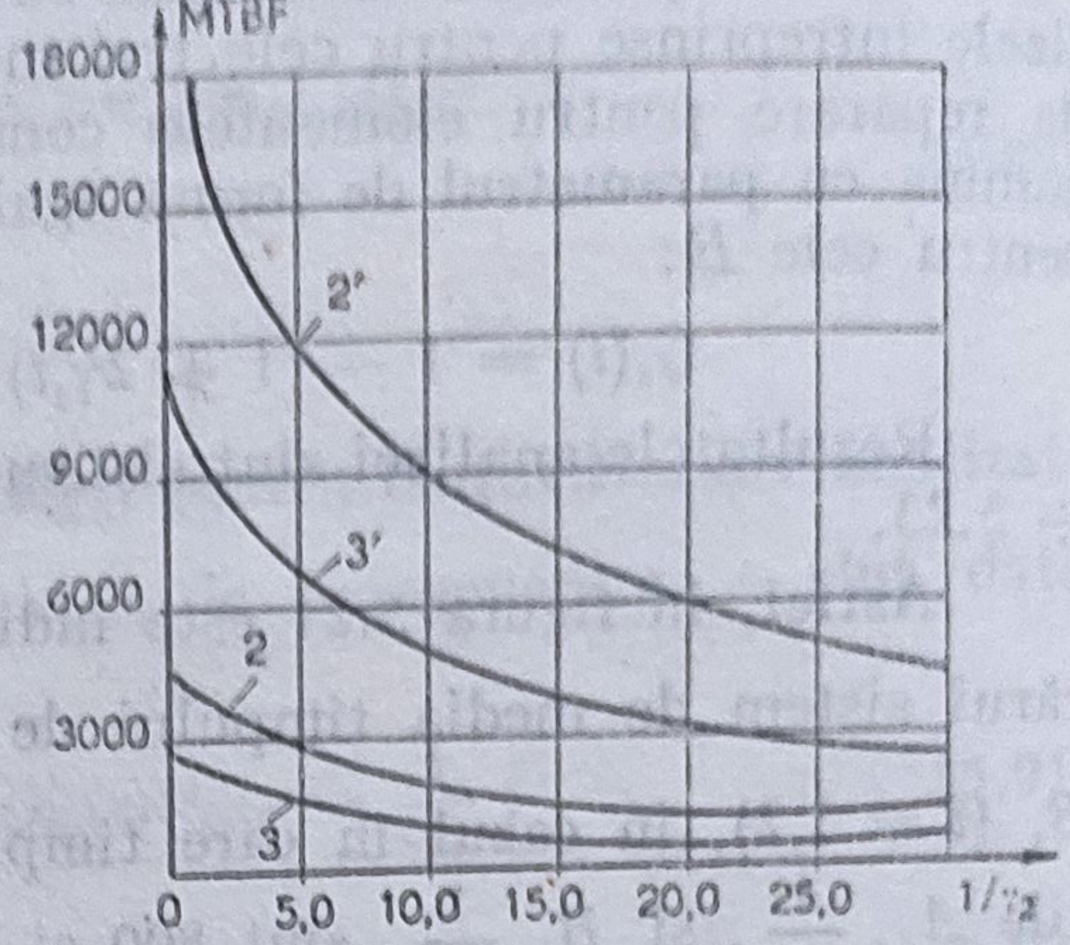


Fig. 5.23. Dependența timpului mediu de funcționare între defectări, MTBF, de media timpului de reparare $1/\gamma_2$ a unităților B_i.

duplex, A, de valoarea coeficientului a. Se constată ușor că performanțele sistemului sînt cu atît mai bune, cu cît rata defectărilor elementelor *off line* este mai mică decît a elementelor *on line* (a este mai apropiat de 0).

Graficele din figura 5.23 prezintă dependența MTBF, corespunzătoare fiecărui sistem, de valoarea mediei timpului de reparare pentru unitățile A_i, $\frac{1}{\gamma_2}$, în două cazuri: atunci cînd mediile timpilor de funcționare $\frac{1}{\lambda_1}$ și $\frac{1}{\lambda_2}$ au valorile 800 și 1 000 ore, respectiv 1 600 și 2 000 ore. Din aceste reprezentări grafice rezultă că cele două sisteme cu structură redondantă au media timpului de funcționare între defectări mult mai mare decît cea a sistemului neredondant și că din nou sistemul duplex ($a = 0$) este mai bun decît sistemul dual. De asemenea se poate constata că pentru sistemul neredondant MTBF nu este dependent de timpul de reparare, pe cînd în ceea ce privește cele două sisteme cu structură redondantă MTBF este cu atît mai mare cu cît $\frac{1}{\gamma_2}$ este mai mic.

Așadar sistemul duplex este mai performant din punctul de vedere al disponibilității și mediei timpului între defectări. Aceste performanțe sînt cu atît mai bune cu cît elementele în așteptare au o rată de defectare mai redusă (cînd $a \to 0$), iar timpii de reparare a elementelor defecte sînt foarte mici.

5.5.3. ANALIZA PERFORMANȚELOR DE FIABILITATE LUÎNDU-SE ÎN CONSIDERARE IMPERFECȚIUNILE RESTABILIRII FUNCȚIONĂRII

Analiza realistă a fiabilității sistemelor cu structură tolerantă la defectări trebuie să țină seama și de imperfecțiunile legate de detecția unor defectări, ca și de cele referitoare la restabilirea funcționării sistemului.

Acestea pot fi modelate prin intermediul factorului de acoperire (*converage*), care poate fi definit ca probabilitatea unui sistem de a se restabili automat la apariția unui defect [5].

În general sistemele tolerante la defectări necesită o funcționare continuă pentru îndeplinirea misiunilor. De aceea, devine necesară aprecierea performanțelor acestor sisteme din punctul de vedere al capacității de mascare a defectărilor la ieșirile sistemului.

Pentru analiza care urmează se acceptă existența unor defecte care nu pot fi detectate de unitățile monitor. De exemplu, pentru sistemul duplex, acestea pot fi defectări de tip intermitent — cînd funcțiile de ieșire variază între 0 și 1, dar nu în acord cu specificările de proiectare. Însă o astfel de defectare poate fi detectată într-un sistem dual de îndată ce ea s-a manifestat, dacă sînt testate ieșirile de la elementele ce funcționează în paralel. Considerînd acest tip de defectare, în continuare se vor evalua coeficientul de disponibilitate și media timpului de funcționare între defectări pentru cele două tipuri de sisteme tolerante la defectări, în vederea comparării lor din punct de vedere fiabilistic. În acest scop se utilizează tot o modelare bazată pe procesele de restabilire Markov, dar se are în vedere și conceptul de acoperire a defectărilor/erorilor.

Față de ipotezele (1) ....(4) prezentate în § 5.5.1 se introduc în plus următoarele ipoteze:

(5) Factorul de acoperire pentru modelul i este c_i, $(i = 1, 2, 3)$;

(6) Atunci cînd unitatea A_i se defectează sau se restabilește, iar comutarea automată nu poate fi executată, timpul de comutare manuală are o valoare finită și este repartizat pentru modelul j după legea $M_{j1}(t)$, cu media $1/\delta_{j1}$, $j = 1, 2, 3$. În mod analog pentru unitățile B_i, $i = 1$ sau 2, se introduce funcția de repartiție M_{j2} cu mediu $1/\delta_{j2}$, $j = 1, 2, 3$;

(7) Duratele de timp cît rămîn nedetectate defectările în unitățile A_i și B_i din sistemul duplex sînt repartizate după o lege exponențială de parametru σ_1 și, respectiv σ_2;

(8) În cazul defectărilor avute în vedere la ipoteza (7) pentru unitățile A_i și B_i, timpii de reparare sînt repartizați după legi avînd funcțiile de repartiție $G_{S1}(t)$, respectiv $G_{S2}(t)$ și cu mediile $1/\gamma_{S1}$ iș, respectiv, $1/\gamma_{S2}$.

Sistemul duplex. Sistemul analizat funcționează după modelul indicat în continuare.

Atunci cînd într-una dintre unitățile A_i sau B_i ale sistemului care funcționează *on line* este detectat un defect, pot avea loc următoarele două evenimente:

(1) restabilire automată — sistemul se restabilește automat cu probabilitatea c_1, comutîndu-se pe rezervă;

(2) restabilirea manuală — sistemul se restabilește manual cu probabilitatea $c_1 = 1 - c_1$.

Pentru cele două cazuri de restabilire considerate anterior unitatea defectă este reparată după realizarea comutării. Atunci cînd este detectat un defect în sistemul de rezervă — aflat în așteptare — unitatea defectă este reparată imediat dacă n-a avut loc nici o comutare.

O analiză a stărilor acestui sistem în condițiile enunțate evidențiază următoarele stări suplimentare față de cele cuprinse în diagrama din figura 5.20;

— starea 1′: unitatea A_i ($i = 1$ sau 2) — care funcționează *on line* — se defectează, iar sistemul se restabilește manual;

— starea 2′: unitatea B_i ($i = 1$ sau 2) — care funcționează *on line* — se defectează, după care sistemul de restabilește manual.

Diagrama tranzițiilor stărilor pentru acest model este indicată în figura 5.24.

Stările 0, 1, 2, 1′, 2′ sînt puncte de regenerare, iar stările 3, 4, 5, 6 — puncte de neregenerare.

Sistemul dual. În acest caz fiecare pereche de unități A_i și B_i ($i = 1, 2$) formează un subsistem al unui calculator. Ambele unități de același tip execută sarcini identice, iar ieșirile lor sînt permanent comparate cu ieșirile testate. Atunci cînd este detectat un defect, unitatea defectă are posibilitatea să fie separată de sistem, în acesta din urmă rămînînd numai o unitate care execută sarcinile *on line*. Se introduce probabilitatea de comutare automată — de acoperire a defectării — c_3, și probabilitatea de comutare manuală, $\bar{c}_3 = 1 - c_3$. După comutare unitatea defectă este reparată.

Se face aceeași ipoteză ca în modelele anterioare (§ 5.5.2), adică atît în cazul sistemelor duplex cît și în cel al sistemelor duale timpul de bună funcționare pentru unitățile componente este repartizat după o lege exponențială de parametri λ_1 și, respectiv, λ_2, corespunzători unităților A_i ($i = 1, 2$) și, respectiv, B_i ($i = 1, 2$).

Așadar, există o categorie de defecte ce nu pot fi detectate cu unitățile monitorului (de exemplu, defectele tranzitorii care înseamnă valoare logică necorespunzătoare) în sistemul duplex, dar pot fi detectate în sistemul dual. Se consideră că aceste defectări apar cu o repartiție exponențială de parametri τ_1 și respectiv, τ_2, corespunzători unităților A_i ($i = 1, 2$), respectiv B_i.

Cu aceste considerente, sistemului dual considerat îi pot fi adăugate — față de stările prezentate în diagrama din figura 5.20 — următoarele stări:

— starea 1″: este detectat un defect tranzitoriu în unitatea A_i; sistemul se restabilește automat;

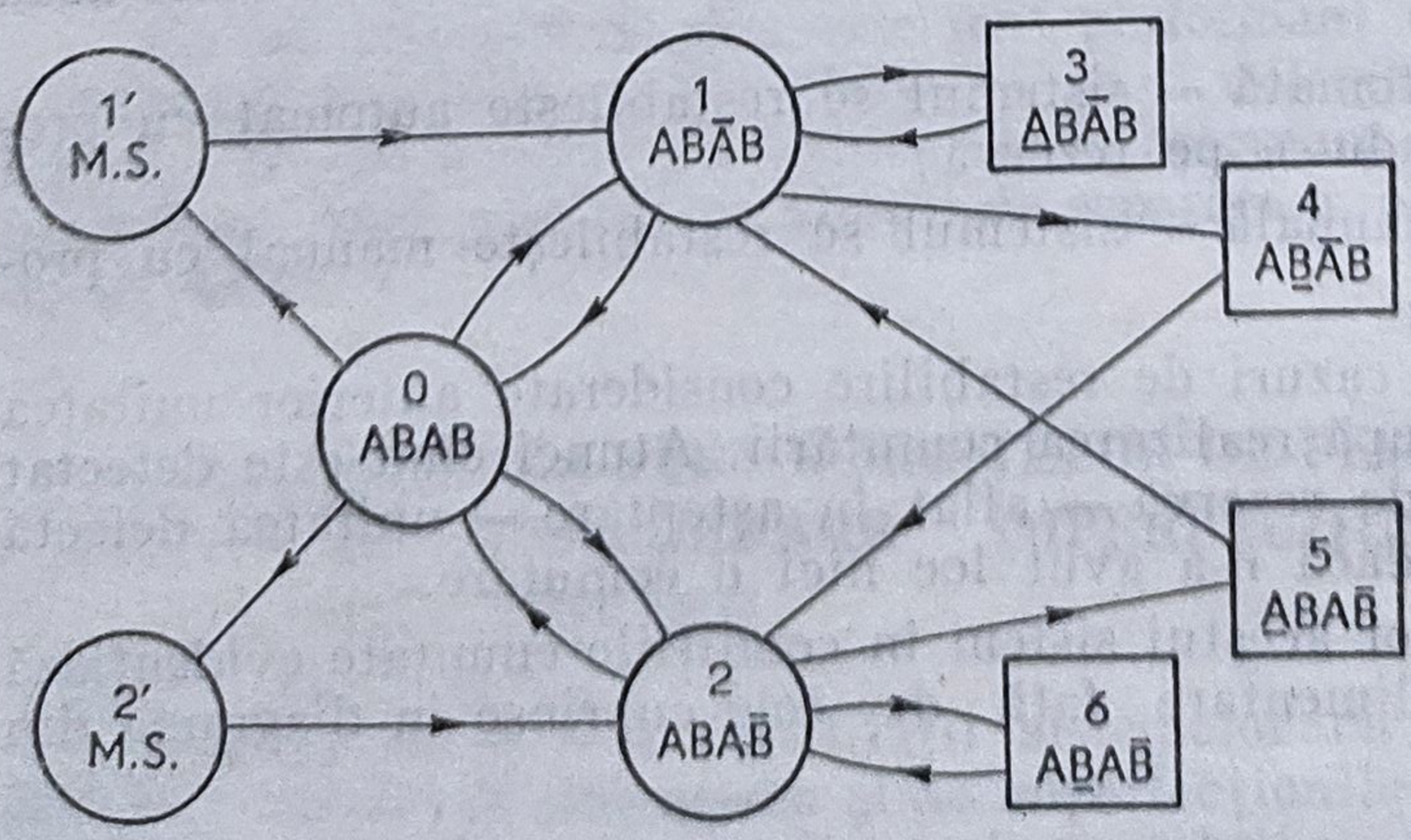


Fig. 5.24. Diagrama tranzițiilor între stări pentru sistemul duplex, în cazul modelării reconfigurării imperfecte.

— starea 2'': este detectat un defect tranzitoriu în unitatea B_i; sistemul se restabilește automat;
— starea 1' : este detectat un defect permanent în unitatea A_i; sistemul se restabilește manual;
— starea 1''': este detectat un defect tranzitoriu în unitatea A_i; sistemul se restabilește manual;
— starea 3' : este detectat un defect permanent în unitatea A_i prin intermediul stării 1''; sistemul se defectează;
— starea 4' : este detectat un defect permanent în unitatea B_i, după ce sistemul a trecut prin starea 1''; sistemul se defectează;
— starea 5' : este detectat un defect permanent în unitatea A_i, după ce sistemul a trecut prin starea 2''; sistemul se defectează;
— starea 6' : este detectat un defect permanent în unitatea B_i, după ce sistemul a trecut prin starea 2''; sistemul se defectează.

Din această enumerare rezultă că stările $i = \{0, 1, 2, 1', 2', 1'', 1''', 2''\}$ sînt puncte de regenerare într-o diagramă a tranzițiilor stărilor, iar stările $i = \{3, 4, 3', 4', 5, 6, 5', 6'\}$ reprezintă puncte de neregenerare. Diagrama tranzițiilor este indicată în figura 5.25.

Calculul coeficientului de disponibilitate și al MTBF pentru sistemul duplex

Se consideră diagrama tranzițiilor între stări dată în figura 5.24. Conform modelului enunțat se pot scrie probabilitățile tranzițiilor $Q_{ij}(t)$, $i \in \mathbf{Rg}_2$, $j \in \mathbf{Z}_2$, precum și $Q^k_{ij}(t)$, $i \in \mathbf{Rg}_2$, $j \in \mathbf{Z}_2$, $k \in \mathbf{NRg}_2$.

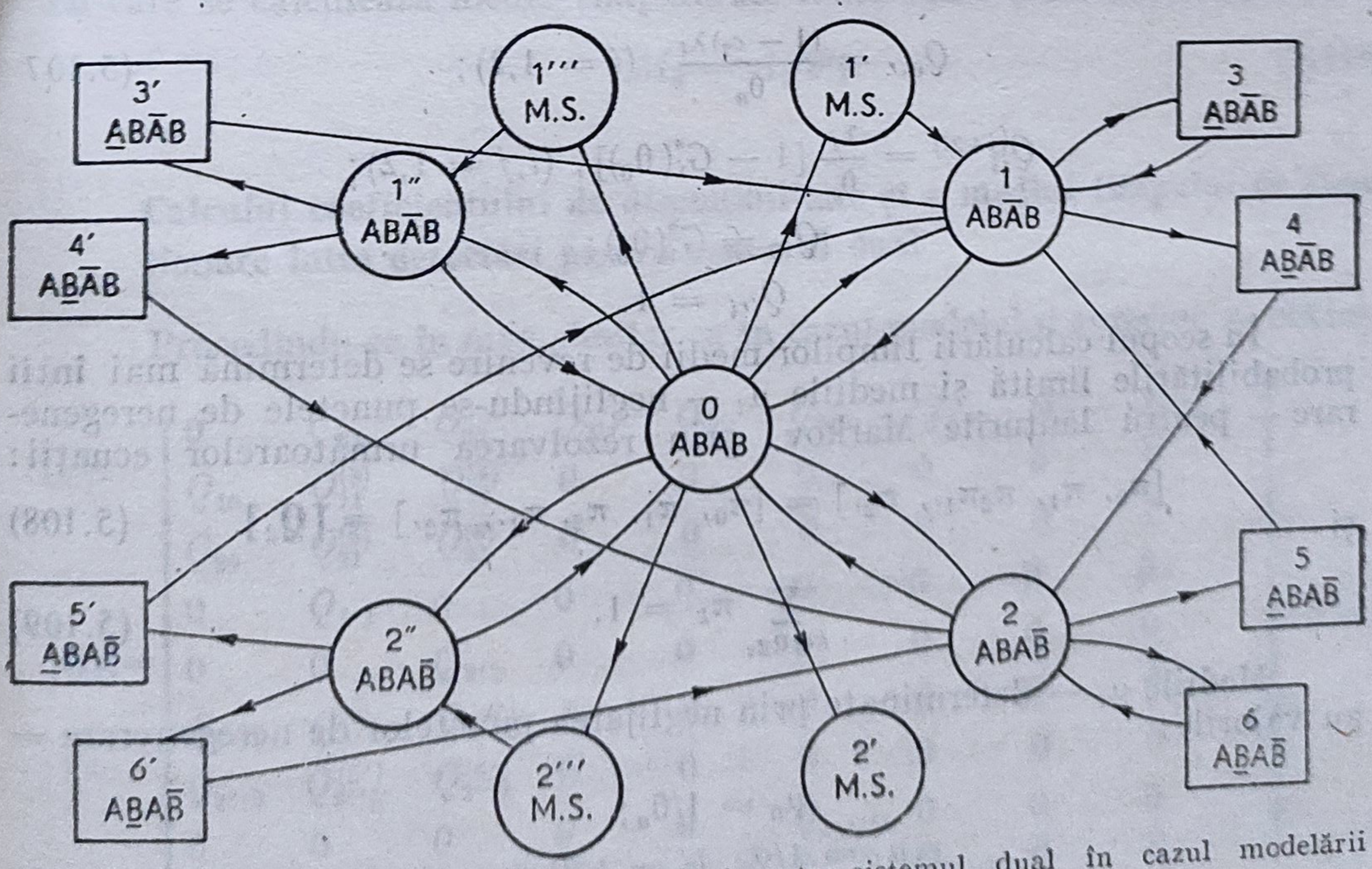


Fig. 5.25. Diagrama tranzițiilor între stări pentru sistemul dual în cazul modelării reconfigurării imperfecte.

Se calculează transformatele Laplace ale acestor expresii rezultînd:

$$Q^*_{0,i}(s) = \frac{\lambda_i + \alpha\lambda_i}{s + \theta_0}, \ (i = 1,2); \tag{5.104}$$

$$Q^*_{0,i}(s) = \frac{(1 - \alpha_0)\lambda_i}{s + \theta_0}; \ (i = 1,2); \tag{5.105}$$

$$Q^*_{i0}(s) = G^*_i(s + \theta_0), \ (i = 1,2); \tag{5.106}$$

$$Q^*_{i,j+2}(s) = \frac{\lambda_j}{s + \theta_0}[1 - G^*_i(s + \theta_0)], \ (i,j = 1,2); \tag{5.107}$$

$$Q^*_{i,i}(s) = M^*_i(s), \ (i = 1,2); \tag{5.108}$$

$$Q^*_{i,j+2}(s) = \frac{\lambda_j}{s + \theta_0}[G^*_i(s) - G^*_i(s + \theta_0)], \ (i,j = 1,2). \tag{5.109}$$

Utilizîndu-se aceste rezultate şi avîndu-se în vedere că stările sînt puncte de regenerare, se poate scrie matricea probabilităţilor de tranziţie [14]:

$$[Q_2] = \begin{bmatrix} 0 & Q_{01} & Q_{02} & Q_{03} & Q_{04} \\ Q_{10} & Q^2_{11} & Q^2_{12} & 0 & 0 \\ Q_{20} & Q^2_{21} & Q^2_{22} & 0 & 0 \\ 0 & Q_{31} & 0 & 0 & 0 \\ 0 & 0 & Q_{42} & 0 & 0 \end{bmatrix} \tag{5.106}$$

unde:

$$Q_{0i} = \frac{(\alpha_0 - \alpha)\lambda_i}{\theta_0}, \ (i = 1,2);$$

$$Q_{0,i+2} = \frac{(1 - \alpha_0)\lambda_i}{\theta_0}, \ (i = 1,2); \tag{5.107}$$

$$Q^{i,j+2}_{ij} = \frac{\lambda_j}{\theta_0}[1 - G^*_i(\theta_0)], \ (i,j = 1,2);$$

$$Q_{i0} = G^*_i(\theta_0);$$

$$Q_{i+1,i} = 1.$$

În scopul calculării timpilor medii de revenire se determină mai întîi probabilităţile limită şi mediile μ_i — neglijîndu-se punctele de neregenerare — pentru lanţurile Markov, prin rezolvarea următoarelor ecuaţii:

$$[\pi_0, \pi_1, \pi_2\pi_{1'}, \pi_{2'}] = [\pi_0, \pi_1, \pi_2, \pi_{1'}, \pi_{2'}] = [Q_2]. \tag{5.108}$$

şi

$$\sum_{i \in Rg_i} \pi_i = 1. \tag{5.109}$$

Mediile μ_i — determinate prin neglijarea punctelor de neregenerare — au valorile:

$$\mu_0 = 1/\theta_0;$$
$$\mu_i = 1/\gamma_i, \ (i = 1,2); \tag{5.110}$$
$$\mu_{i'} = = 1/\beta_{i'}, \ (i' = 1,2).$$

Pe baza determinărilor anterioare se pot calcula timpii medii de revenire, l_{ii}:

$$l_{ii} = \frac{\sum_{k \in Rg_2} \pi_{k|\mu_k}}{\pi_i}, \quad i \in Rg_2. \tag{5.111}$$

Mediile necondiționale — calculate fără a mai neglija punctele de neregenerare — sînt date de ecuațiile:

$$\zeta_0 = \mu_0 = 1/\theta_a;$$

$$\zeta_i = \frac{1}{\theta_0}(1 - G_i^*(\theta_0)), \quad (i = 1,2). \tag{1.112}$$

Cu aceste determinări se pot recalcula probabilitățile limită pentru stările de operare:

$$p_i = \zeta_i / l_{ii}, \quad (i \in A_2), \tag{5.113}$$

iar de aici se determină imediat coeficientul de disponibilitate:

$$A_2 = \sum_{k \in A_2} p_k. \tag{5.114}$$

În continuare se obține numărul așteptat de defectări pe unitatea de timp a sistemului, acesta aflîndu-se în starea sigură:

$$\mathcal{F}_2 = \sum_{k=1}^{2} \left[\frac{Q_{k,1+2k} + Q_{k,2+2k}}{l_{kk}} + \frac{1}{l_{k'k'}} \right]. \tag{5.115}$$

cu care se calculează media timpului de funcționare între defectări:

$$M_2 = A_2 / \mathcal{F}_2. \tag{5.116}$$

Calculul coeficientului de disponibilitate și a mediei timpului de funcționare între defectări pentru sistemul dual

Procedîndu-se în mod similar ca în cazul modelului anterior se obține:

$$[Q_3] = \begin{bmatrix}
0 & Q_{01} & Q_{02} & Q_{01'} & Q_{02'} & Q_{01''} & Q_{02''} & Q_{01'''} & Q_{02'''} \\
Q_{10} & Q_{11}^{(3)} & Q_{12}^{(4)} & 0 & 0 & 0 & 0 & 0 & 0 \\
Q_{20} & Q_{21}^{(5)} & Q_{22}^{(6)} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & Q_{1'1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & Q_{2'2} & 0 & 0 & 0 & 0 & 0 & 0 \\
Q_{1''0} & Q_{1''1}^{(3')} & Q_{1''2}^{(4')} & 0 & 0 & 0 & 0 & 0 & 0 \\
Q_{2''0} & Q_{2''1}^{(5')} & Q_{2''2}^{(6')} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & Q_{1'''1'''} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & Q_{2'''2'''} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & & & &
\end{bmatrix} \tag{5.117}$$

unde:

$$Q_{0i} = \frac{2c_s\lambda_i}{0_S}, \qquad (i = 1,2);$$

$$\begin{aligned}
Q_{0i'} &= \frac{2\bar{c}_s\lambda_i}{0_S}, && (i = 1,2);\\
Q_{0i''} &= \frac{2c_s\sigma_i}{0_S}, && (i = 1,2);\\
Q_{i0} &= G_i^*(0_0), && (i = 1,2);\\
Q_{i,j}^{j+2i} &= \frac{\lambda_j}{0_0}(1 - G_i^*(0_0)), && (i,j = 1,2);\\
Q_{i'i} &= 1, && (i = 1,2);\\
Q_{i'0} &= G_{Si}^*(0_0), && (i = 1,2);\\
Q_{i',j}^{(j+2i)'} &= \frac{\lambda_j}{0_0}(1 - G_{Si}^*(0_0)), && (i,j = 1,2);\\
Q_{i',i''} &= 1, && (i = 1,2).
\end{aligned} \tag{5.118}$$

În continuare se rezolvă ecuaţiile:

$$[\pi_0\pi_1\pi_2\pi_1, \pi_2, \pi_1,,\pi_2,,\pi_1,,,\pi_2,,,] = [\pi_0\ \pi_1\ \pi_2\ \pi_1,\ \pi_2,\pi_1,,\pi_2,,\pi_1,,,\pi_2,,,]\ \mathbf{Q}_3 \tag{5.119}$$

şi

$$\sum_{k\in \mathbf{Rg}_3} \pi_k = 1; \tag{5.120}$$

rezultă mediile μ_i, $i \in \mathbf{Rg}_3$, obţinute prin neglijarea punctelor de neregenerare:

$$\begin{aligned}
\mu_0 &= 1/0_S;\\
\mu_i &= 1/\gamma_i, && (i = 1,2);\\
\mu_{i''} &= 1/\gamma_{Si}, && (i = 1,2);\\
\mu_{i'} &= \mu_{i''} = 1/\delta_{3i}, && (i = 1,2).
\end{aligned} \tag{5.121}$$

Pe baza valorilor π_k şi μ_i — determinate prin rezolvarea ecuaţiilor (5.119), (5.120) şi, respectiv, (5.121) — se calculează timpii medii de revenire, l_{ii}:

$$l_{ii} = \sum_{k\in \mathbf{Rg}_3} \pi_k\mu_k/\pi_i, \quad i \in \mathbf{Rg}_3. \tag{5.122}$$

De asemenea, în funcţie de rezultatele anterioare se obţin mediile ζ_i ($i \in \mathbf{A}_3$), fără a mai neglija punctele de neregenerare:

$$\begin{aligned}
\zeta_0 &= \mu_0;\\
\zeta_{i''} &= \frac{1}{0_0}(1 - G_{Si}^*(0_0)), && (i = 1,2);\\
\zeta_i &= \frac{1}{0_0}(1 - G_i^*(0_0)), && (i = 1,2),
\end{aligned} \tag{5.123}$$

cu care se pot calcula probabilităţile p_i, $i \in \mathbf{A}_3$:

$$p_i = \zeta_i / l_{ii}, \quad i \in \mathbf{A}_3. \tag{5.124}$$

Utilizîndu-se aceste probabilităţi se determină coeficientul de disponibilitate pentru sistemul analizat:

$$A_3 = \sum_{k \in \mathbf{A}_3} p_k. \tag{5.125}$$

Mai departe, se obţine numărul aşteptat de defectări ale sistemului:

$$\mathcal{F}_3 = \sum_{i=1}^{2} \left(\frac{Q_{i,1+2i} + Q_{i,2+2i}}{l_{ii}} + \frac{1}{l_{i'i'}} + \right.$$

$$\left. + \frac{Q_{i''(1+2i)'} + Q_{i''(2+2i)''}}{l_{i''i''}} + \frac{1}{l_{i'''i'''}} \right), \tag{5.126}$$

unde:

$$\begin{aligned} Q_{i,i+2i} &= Q_{i1}^{(1+2i)}, \quad (i = 1, 2); \\ Q_{i,2+2i} &= Q_{i2}^{(2+2i)}, \quad (i = 1, 2); \\ Q_{i''(1+2i)'} &= Q_{i''1}^{(1+2i)'} \quad (i = 1, 2); \\ Q_{i''(2+2i)'} &= Q_{i''2}^{(2+2i)'} \quad (i = 1, 2). \end{aligned} \tag{5.127}$$

Cu aceste rezultate se obţine media timpului de funcţionare între defectări,

$$M_3 = A_3 / \mathcal{F}_3. \tag{5.128}$$

Exemplul 5.5. Se consideră că timpii de reparare au aceeaşi lege de repartiţie, legea gamma cu parametrul de formă 2 şi media timpului de repararea $\frac{1}{\gamma_i}$, atunci cînd în sistem apar defectări de tip permanent:

$$G_i(t) = 1 - (1 + 2\gamma_i t) \exp(-2\gamma_i t), \ (i = 1, 2) \tag{5.129}$$

şi o lege gamma cu parametrul de formă 2, media timpului de reparare fiind $\frac{1}{\gamma_{si}}$, cînd în sistem sînt detectate defectări tranzitorii:

$$G_{si}(t) = 1 - (1 + 2\gamma_{si} t) \exp(-2\gamma_{si} t), \ (i = 1, 2). \tag{5.130}$$

Calculîndu-se coeficientul de disponibilitate în condiţiile:

$$c_2 = c_3;$$

$$\delta_{21} = \delta_{22} = \delta_{31} = \delta_{32},$$

iar

$\frac{1}{\delta_{ij}} = 0{,}1$ ore, respectiv 1 oră, $(i = 2{,}3; \ j = 1{,}2)$,

$$\frac{1}{\lambda_1} = 800 \text{ ore}, \quad \frac{1}{\lambda_2} = 1\,000 \text{ ore};$$

$$\frac{1}{\gamma_1} = 5 \text{ ore}, \quad \frac{1}{\gamma_2} = 8 \text{ ore},$$

$$\frac{1}{\sigma_1} = \frac{1}{\sigma_2} = 2\,000 \text{ ore}$$

și

$$\frac{1}{\gamma_{S1}} = \frac{1}{\gamma_{S2}} = 0{,}1 \text{ ore.}$$

acesta apare ca o funcție de coeficientul de acoperire c (fig. 5.26).

Se observă că disponibilitatea sistemului duplex este mai ridicată decît cea a sistemului dual, în aceleași condiții de performanță pentru elementele componente, și cu cît valoarea parametrului $1/\vartheta_{ij}$ este mai mică, cu atît coeficientul de disponibilitate este mai mare.

Determinîndu-se MTBF în condițiile:

$c_2 = c_3 = c$, $\frac{1}{\lambda_1} = 800$ ore, $\frac{1}{\lambda_2} = 1\,000$ ore, $\frac{1}{\gamma_1} = 5$ ore, $\frac{1}{\gamma_2} = 8$ ore, $\frac{1}{\sigma_1} = \frac{1}{\sigma_2} = 2\,000$ ore, $\frac{1}{\gamma_{S1}} = \frac{1}{\gamma_{S2}} = 0{,}1$ ore și $\frac{1}{\delta_{ij}} = 0{,}1$ ore ($i = 2{,}3$ și $j = 1{,}2$), poate fi evidențiată dependența acestui indicator de fiabilitate de factorul de acoperire c (fig. 5.27).

Pe baza acestor rezultate se poate aprecia că media timpului de funcționare între defectări, MTBF, a sistemului duplex are o valoarea mai mare decît a sistemului dual.

În figurile 5.26 și 5.27 se indică dependența lui A, respectiv MTBF, de factorul c, în ipoteza că acest factor ar fi același pentru ambele sisteme.

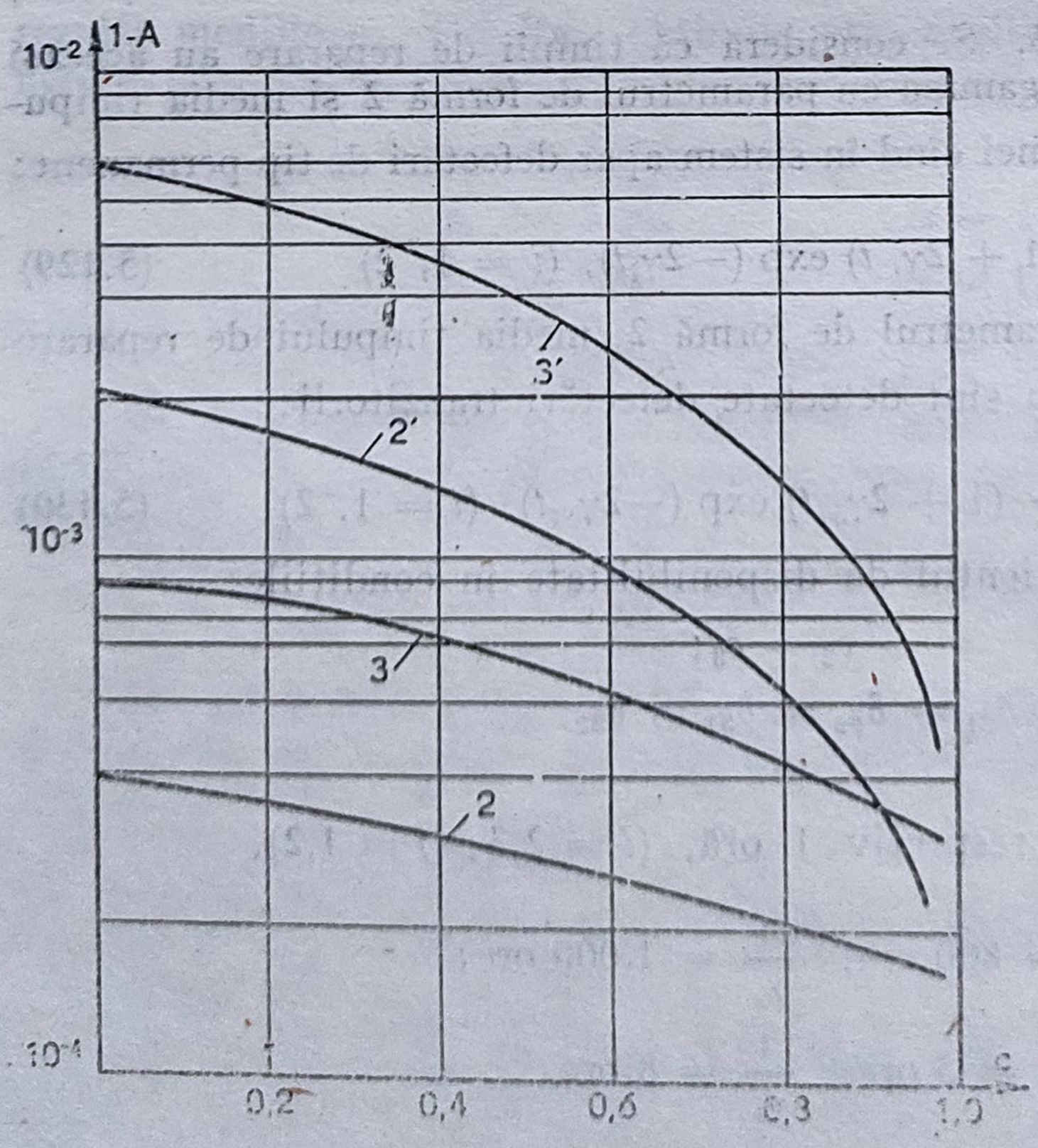


Fig. 5.26. Dependența coeficientnlui de indisponibilitate pentru sistemele duplex (curbele 2, 2′)și dual (curbele 3, 3′) de factorul de acoperire, c.

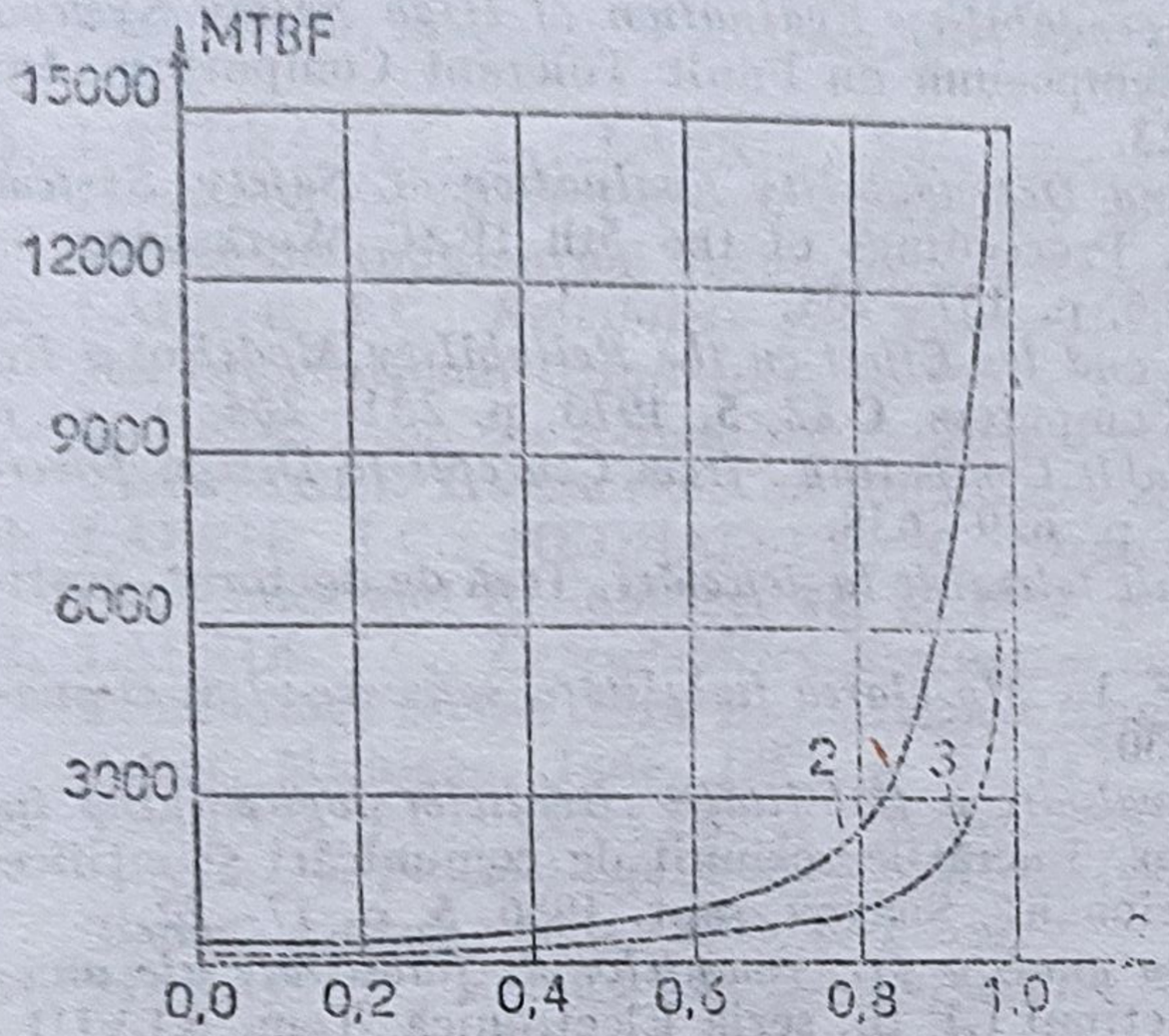


Fig. 5.27. Dependența MTBF pentru sistemele duplex (curba 2) și dual (curba 3) de factorul de acoperire, c.

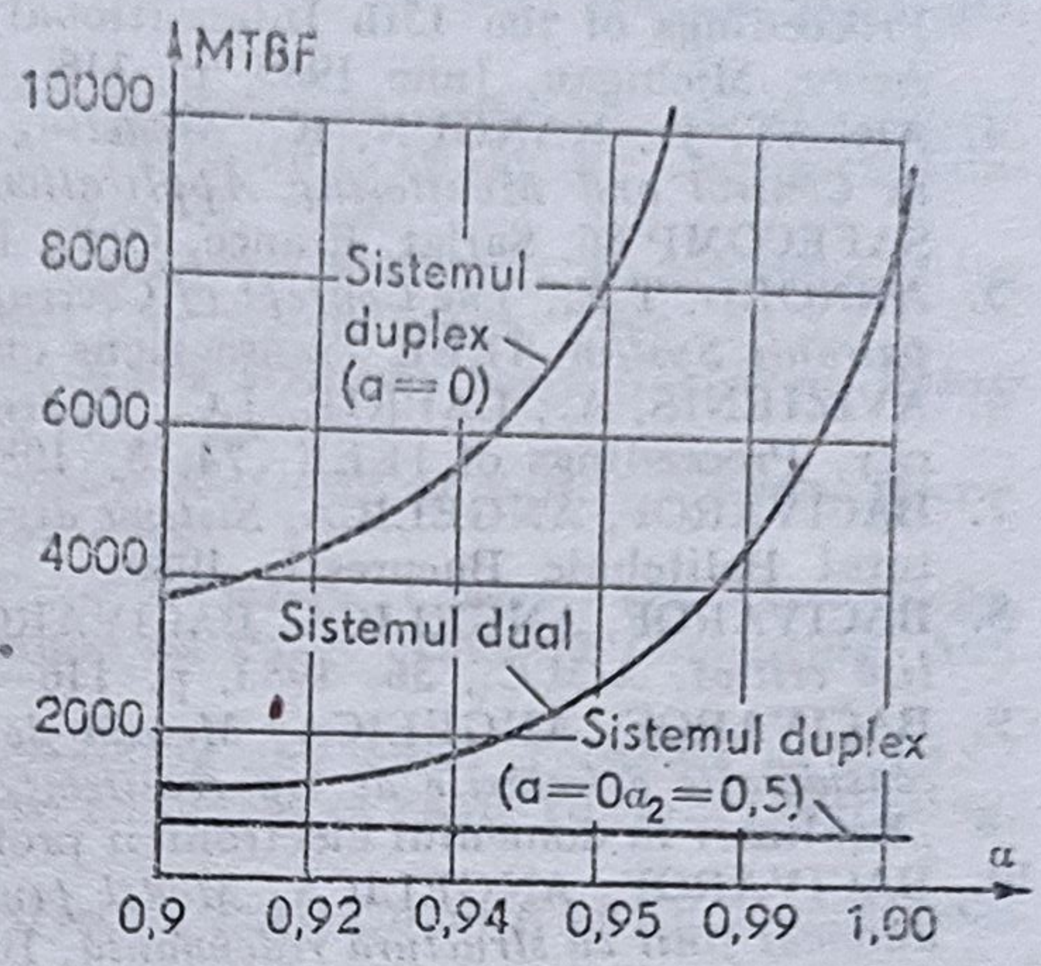


Fig. 5.28. Dependența MTBF pentru sistemele analizate de factorul de acoperire, c.

De fapt, întrucît mecanismele de comutare a unității defecte sînt diferite, factorul de acoperire nu este același pentru cele două sisteme. În mod real, este dificil pentru sistemul duplex să execute o comutare automată cînd cele două module — *on line* și respectiv *off line* — procesează sarcini diferite, fără a îngloba în structura acestora programe speciale care să detecteze starea lor de defectare. În cazul sistemului dual modulele funcționale execută sarcini *on line* identice, iar ieșirile sînt comparate. Cînd oricare dintre aceste module se defectează, sistemul își continuă funcționarea, separîndu-se numai modulul defect. Deci sistemul dual are factorul de acoperire mai mare.

În figura 5.28 se evidențiază dependența timpului mediu de funcționare între defectări, MTBF, pentru sistemul dual de factorul de acoperire c_3, cînd acesta are valori mari, și se indică de asemenea valoarea timpului mediu de funcționare între defectări pentru sistemul duplex, atunci cînd $c_2 = 0{,}5$ (o valoare realistă). Cu luarea în considerare a mecanismelor de comutare diferite se poate evidenția în acest mod că sistemul dual are MTBF mai mare decît sistemul duplex.

În paragraful anterior, fără a ține seama de imperfecțiunile legate de comutarea rezervelor, s-au obținut pentru sistemul duplex performanțe superioare referitoare la disponibilitate și media timpului de funcționare între defectări comparativ cu sistemul dual. Acum, în condițiile unei analize mai realiste — cu luarea în considerare și a imperfecțiunilor comutării rezervelor — a fost evidențiat un rezultat diferit: sistemul dual are o medie a timpului de funcționare între defectări mai mare decît sistemul duplex.

BIBLIOGRAFIE

1. ARLAT, J.; LAPRIE, J.C., *Performance Related Dependability Evaluation of Supercomputer Systems*, Microelectronics and Reliability, 24, 4, aug. 1984, p. 717–742.
2. ARLAT, J.; BLANQUART, J.P.; LAPRIE, J.C., *On the Certification of Computing Systems : the EVE Projet*, Proceedings of the 4th International Conference on Reliability and Maintainability, Perros-Guirec, France, May 1984, p. 650–656 .

ANEXA

INDICATORI DE FIABILITATE

A.1. Indicatorii de fiabilitate sînt mărimi care estimcază cantitativ fiabilitatea produselor*.

Datorită caracterului aleator al factorilor care influențează funcționarea produselor și procesele de apariție, dezvoltare și înlăturare a defectărilor, estimarea cantitativă a fiabilității implică utilizarea metodelor teoriei probabilităților și ale statisticii matematice.

Se definesc *indicatori generali de fiabilitate*, care pot fi utilizați pentru toate categoriile de produse și *indicatori de fiabilitate caracteristici produselor reparabile* (cu reînnoire). În cazul sistemelor de telecomunicații, de calcul, informaționale etc. care conțin pe lîngă partea materială (de hardware) și o parte logică (de software) este utilă folosirea unor *indicatori de fiabilitate specifici software-ului*.

La alegerea indicatorilor de fiabilitate ai unui produs se va avea în vedere, în principal, *caracterul produsului* (reparabil sau nereparabil, material sau logic etc.), precum și importanța și caracterul misiunii pe care acesta o are de îndeplinit (produse de mare răspundere funcțională sau de uz curent, produse pentru utilizare de scurtă sau lungă durată, funcționare continuă sau intermitentă etc.).

Simbolurile și definițiile matematice ale *indicatorilor generali de fiabilitate* sînt sistematizate în tabelul A.1, iar relațiile dintre ei sînt prezentate în tabelul A.2.

Principalii *indicatori de fiabilitate specifici produselor reparabile* sînt definiți în tabelul A.3.

A.2. Exprimarea analitică a indicatorilor de fiabilitate ai produselor se bazează pe adoptarea unei anumite legi de repartiție a timpului de funcționare fără defectare.

În tabelul A.4 sînt prezentate principalele legi teoretice de repartiție ce pot fi utilizate pentru caracterizarea fiabilității componentelor și sistemelor electronice și de telecomunicații.

---

* În cele ce urmează termenul generic de *produs* desemnează orice componentă, subsistem (bloc funcțional) sau sistem (echipament) supus observațiilor, ce poate fi considerat de sine stătător și poate fi încercat în mod separat.

Tabelul A.1. SIMBOLURILE ȘI DEFINIȚIILE MATEMATICE ALE INDICATORILOR GENERALI DE FIABILITATE

Nr. crt.	Indicator de fiabilitate	Simbol	Definiție	Unitatea de măsură
1	Funcția de repartiție a timpului de funcționare (funcție de nonfiabilitate)	$F(t)$	Probabilitatea ca un produs să se defecteze în intervalul $(0, t)$: $F(t) = P(T \leqslant t)$	—
2	Densitatea de probabilitate a timpului de funcționare	$f(t)$	Limita raportului dintre probabilitatea de defectare în intervalul $(t, t + \Delta t)$ și mărimea intervalului, cînd $\Delta t \to 0$ $f(t) = \lim_{\Delta t \to 0} \frac{P(t \leqslant T \leqslant t + \Delta t)}{\Delta t}$	ore^{-1}
3	Funcția de fiabilitate*	$R(t)$	Probabilitatea ca un produs să funcționeze fără defectare în intervalul $(0, t)$, în condiții determinate: $R(t) = P(T > t)$	—
4	Rata (intensitatea) de defectare**	$z(t)$	Limita raportului dintre probabilitatea de defectare în intervalul $(t, t + \Delta t)$, condiționată de buna funcționare în intervalul $(0, t)$ și mărimea intervalului Δt, cînd $\Delta t \to 0$ $z(t) = \lim_{\Delta t \to 0} \frac{P(t < T \leqslant t + \Delta t \mid T > t)}{\Delta t}$	ore^{-1}
5	Media timpului de funcționare*** — MTTF — MTFF — MTBF	m	Valoarea medie a timpului de funcționare: $m = \int_0^\infty t\, f(t)\, dt$	ore
6	Dispersia timpului de funcționare	D	Momentul centrat de ordinul 2 al timpului de funcționare: $D = \int_0^\infty (t - m)^2 f(t)\, dt$	ore^2
7	Abaterea medie pătratică a timpului de funcționare	σ	$\sigma = \sqrt{D}$	ore
8	Cuantila timpului de funcționare	t_F	Timpul în care un produs funcționează cu o anumită probabilitate $(1 - F)$, $P(T \leqslant t_F) = F$	ore

* Funcția de fiabilitate în intervalul (t_1, t_2) se definește prin relația:

$$R(t_1, t_2) = \frac{R(t_2)}{R(t_1)}.$$

** Rata medie de defectare în intervalul (t_1, t_2) se definește ca raportul dintre probabilitatea de defectare în intervalul (t_1, t_2), condiționată de buna funcționare în intervalul $(0, t_1)$, și de mărimea intervalului $(t_2 - t_1)$:

$$\bar{z}(t_1, t_2) = \frac{P(t_1 < T \leqslant t_2 \mid T > t_1)}{t_2 - t_1}.$$

*** *Media timpului de funcționare* este momentul de ordinul 1 al variabilei aleatoare timp de funcționare. Ea reprezintă *valoarea medie a timpului de funcționare pînă la defectare*, în cazul produselor nereparabile (MTTF – *M*ean *T*ime *T*o *F*ailure, utilizînd terminologia din literatura tehnică anglo-saxonă), sau *pînă la prima defectare*, în cazul produselor reparabile (MTFF – *M*ean *T*ime *T*o *F*irst *F*ailure). Dacă repararea poate fi asimilată cu înlocuirea, reprezintă *valoarea medie a timpului de funcționare între două defectări succesive* (MTBF – *M*ean *T*ime *B*etween *F*ailures).

Tabelul A.2

Nr. crt.	Indicator	Exprimat în funcție de indicatorul $F(t)$	$f(t)$	$R(t)$	$z(t)$
1	$F(t)$	–	$\int_0^t f(u)\mathrm{d}u$	$1 - R(t)$	$1 - \exp\left[-\int_0^t z(u)\mathrm{d}u\right]$
2	$f(t)$	$\frac{\mathrm{d}F(t)}{\mathrm{d}t}$	–	$-\frac{\mathrm{d}R(t)}{\mathrm{d}t}$	$z(t) \cdot \exp\left[-\int_0^t z(u)\mathrm{d}u\right]$
3	$R(t)$	$1 - F(t)$	$\int_t^\infty f(u)\mathrm{d}u$	–	$\exp\left[-\int_0^t z(u)\mathrm{d}u\right]$
4	$z(t)$	$\frac{1}{1 - F(t)} \cdot \frac{\mathrm{d}F(t)}{\mathrm{d}t}$	$\frac{f(t)}{\int_t^\infty f(u)\mathrm{d}u}$	$-\frac{1}{R(t)} \cdot \frac{\mathrm{d}R(t)}{\mathrm{d}t}$	–
5	m	$\int_0^\infty [1 - F(t)]\mathrm{d}t$	$\int_0^\infty t\, f(t)\mathrm{d}t$	$\int_0^\infty R(t)\mathrm{d}t$	$\int_0^\infty \exp\left[-\int_0^t z(u)\mathrm{d}u\right]\mathrm{d}t$

OBSERVAȚIE. Prezintă interes și următoarele relații de legătură între indicatorii generali de fiabilitate:

$$z(t) = \frac{f(t)}{R(t)} = \frac{f(t)}{1 - F(t)};$$

$$\sigma^2 = D = \int_0^{00} t^2 f(t)\mathrm{d}t - m^2$$

Tabelul A.3

Nr. crt.	Indicator	Simbol	Definiţie	Unitatea de măsură
1	Funcţia de repartiţie a timpului de reparare	$F_{rep}(t)$	Probabilitatea ca produsul să fie reparat într-un interval de timp de mărimea t: $F_{rep}(t) = P(T_{rep} \leqslant t)$	—
2	Densitatea de probabilitate a timpului de reparare	$f_{rep}(t)$	Limita raportului dintre probabilitatea ca durata de reparare (T_{rep}) a produsului să fie cuprinsă în intervalul $(t, t + \Delta t)$ şi mărimea intervalului, cînd $\Delta t \to 0$; $f_{rep}(t) = \lim\limits_{\Delta t \to 0} \frac{P(t < T_{rep} \leqslant t + \Delta t)}{\Delta t}$	—
3	Media timpului de reparare (MTR)	m_{rep}	Valoarea medie a timpului de reparare: $m_{rep} = \int_0^\infty t f_{rep}(t)dt$	ore
4	Numărul mediu de intrări în funcţiune în intervalul $(0, t)$	$H_2(t)$	Valoarea medie a numărului de intrări în funcţiune în intervalul $(0, t)$: $H_2(t) = M(N_{2t})$	—
5	Numărul mediu de defectări în intervalul $(0, t)$	$H_1(t)$	Valoarea medie a numărului de defectări în intervalul $(0, t)$: $H_1(t) = M(N_{1t})$	—
6	Densitatea intrărilor în funcţiune	$h_2(t)$	Limita raportului dintre probabilitatea uneia sau mai multor intrări în funcţiune în intervalul $(t, t + \Delta t)$ şi mărimea intervalului, cînd $\Delta t \to 0$: $h_2(t) = \lim\limits_{\Delta t \to 0} \frac{P(t, t + \Delta t)}{\Delta t}$	ore^{-1}
7	Densitatea defectărilor	$h_1(t)$	Limita raportului dintre probabilitatea uneia sau mai multor defectări, în intervalul $(t, t + \Delta t)$ şi mărimea intervalului, cînd $\Delta t \to 0$: $h_1(t) = \lim\limits_{\Delta t \to 0} \frac{P(t, t + \Delta t)}{\Delta t}$	ore^{-1}
8	Disponibilitatea	$A(t)$	Probabilitatea ca produsul să fie în stare de funcţionare la momentul t: $A(t) = R(t) + \int_0^t h_2(\tau)[1 - F_{rep}(t-\tau)]d\tau$	—

Tabelul A.3 (*continuare*)

Nr. crt.	Indicator	Simbol	Definiție	Unitatea de măsură
9	Coeficientul de disponibilitate	A	Raportul dintre media timpului de funcționare și valoarea medie a unui ciclu de funcționare-reparare: $A = \frac{m}{m_{rep} + m}$	—

OBSERVAȚII:

1. Simbolurile utilizate au următoarele semnificații:

T_{rep} — durata de reparare a unui produs;

N_{1t} — numărul de defectări în intervalul $(0, t)$;

N_{2t} — numărul de intrări în funcțiune în intervalul $(0, t)$.

2. Indicatorii de la nr. crt. 6 și 7 se definesc în funcție de cei de la nr. crt. 4 și respectiv 5 prin următoarele relații:

$$h_2(t) = \frac{dH_2(t)}{dt};$$

$$h_1(t) = \frac{dH_1(t)}{dt}.$$

3. Formula pentru calculul disponibilității (nr. crt. 8) este adevărată în cazul cînd produsul este nou la momentul $t = 0$.

Tabelul A.4

Indicator	Legea de repartiție a timpului de funcționare: exponențială	Weibull	normală	log-normală	Observații
$F(t)$	$1-\exp(-\lambda t)$	$1-\exp\left[-\left(\frac{t-\gamma}{\eta}\right)^{\beta}\right]$	$\Phi\left(\frac{t-m_0}{\sigma_0}\right)$	$\Phi\left(\frac{\ln t-m_0}{\sigma_0}\right)$	$\Phi(x)=\int_{-\infty}^{x}\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{u^2}{2}\right)du$ (funcția Laplace)
$f(t)$	$\lambda\exp(-\lambda t)$	$\frac{\beta(t-\gamma)^{\beta-1}}{\eta^{\beta}}\exp\left[-\left(\frac{t-\gamma}{\eta}\right)^{\beta}\right]$	$\frac{1}{\sigma_0\sqrt{2\pi}}\exp\left[-\frac{1}{2}\left(\frac{t-m_0}{\sigma_0}\right)^2\right]$	$\frac{1}{t\sigma_0\sqrt{2\pi}}\exp\left[-\frac{1}{2}\left(\frac{\ln t-m_0}{\sigma_0}\right)^2\right]$	$\eta^{\beta}=\alpha$ (parametru de scară)
$R(t)$	$\exp(-\lambda t)$	$\exp\left[-\left(\frac{t-\gamma}{\eta}\right)^{\beta}\right]$	$\Phi\left(\frac{m_0-t}{\sigma_0}\right)$	$\Phi\left(\frac{m_0-\ln t}{\sigma_0}\right)$	—
$z(t)$	λ	$\frac{\beta(t-\gamma)^{\beta-1}}{\eta^{\beta}}$	$\frac{\frac{1}{\sigma_0\sqrt{2\pi}}\exp\left[-\frac{1}{2}\left(\frac{t-m_0}{\sigma_0}\right)^2\right]}{\Phi\left(\frac{m_0-t}{\sigma_0}\right)}$	$\frac{\frac{1}{\sigma_0\sqrt{2\pi}}\exp\left[-\frac{1}{2}\left(\frac{\ln t-m_0}{\sigma_0}\right)^2\right]}{\Phi\left(\frac{m_0-\ln t}{\sigma_0}\right)}$	—
m	$\frac{1}{\lambda}$	$\gamma+\eta\Gamma\left(\frac{1}{\beta}+1\right)$	m_0	$\exp\left(m_0+\frac{\sigma_0^2}{2}\right)$	$\Gamma(z)=\int_0^{\infty}t^{z-1}\exp(-t)dt$ (funcția Euler de speța 1)
D	$\frac{1}{\lambda^2}$	$\eta^2\left[\Gamma\left(\frac{2}{\beta}+1\right)-\Gamma^2\left(\frac{1}{\beta}+1\right)\right]$	σ_0^2	$\exp(2m_0+\sigma_0^2)\left[\exp(\sigma_0^2)-1\right]$	—
σ	$\frac{1}{\lambda}$	$\eta\sqrt{\Gamma\left(\frac{2}{\beta}+1\right)-\Gamma^2\left(\frac{1}{\beta}+1\right)}$	σ_0	$\exp\left(m_0+\frac{\sigma_0^2}{2}\right)\sqrt{\exp(\sigma_0)^2-1}$	—
t_p	$\frac{1}{\lambda}\ln\frac{1}{1-F}$	$\gamma+\eta\sqrt[\beta]{\ln\frac{1}{1-F}}$	$m_0+\Phi^{-1}(F)\sigma_0$	$\exp\left[m_0+\Phi^{-1}(F)\sigma_0\right]$	$\Phi^{-1}(F)$ (inversa funcției Laplace)

HIGH RELIABILITY ELECTRONIC STRUCTURES FAULT TOLERANCE

— Abstract —

Fault tolerance is a main approach to ensuring a high reliability of computing, traffic-control systems, nuclear plants and military appli-cations. This means the accurate performance of a set of specified operations in the presence of failures, without any external interference.

System reliability is generally attained by using structural redundancy that ensures continuos system operation, while its availability may be also obtained by system testing in normal uninterrupted operation.

The book is made up of five sections.

The first one presents the fault tolerance domain and its general concepts.

The next section analyses the redundant structures that may be applied to hardware systems to protect them against failures. The optimal solutions concerning the implementation of different redundant structures are also investigated.

Some aspects regarding the use of error detection techniques in fault systems are discussed in the third section. A special attention is paid to the aspects concerning the implementation of self-checking systems.

In the next section, the sophisticated methods for fault tolerance implementation, including those based on self-recovery and software approaches are analysed.

It is worth mentioning the model of optimal recovery in multiprocessor networks, as well as the studies regarding recovery blocks, N-version and N-self-checking programming.

The last section of the book is concerned with the reliability evaluation of some fault tolerant structures, using adequate mathematical models and specific reliability indices.

In order to efficiently solve the complex problems related to the study of fault tolerant systems, different mathematical methods based on the probability theory, boolean algebra, graphs theory, Markov processes a.o. have been used.

The great number of examples and case studies presented in the book results in an adequate balance between the theoretical approach and the practical applications.

The present work may be used as a text-book for students, practitioners and researchers of fault tolerance. Firstly, it provides a starting point for fault tolerant system design. Secondly, the principles may be applied in order to examine and develop existing systems, providing them with fault tolerance possibilities.

CUPRINS

CONTENS

Chapter 5. RELIABILITY EVALUATION OF FAULT TOLERANT SYSTEMS

Redactor : maior dr. ing. AL. MIHALCEA
Tehnoredactor : D. ANDREI

Bun de tipar : 31.07.1989. Apărut 1989.
Coli de tipar 17¾. B 10288

Tiparul executat sub comanda nr. 506
la I. P. „Filaret", str. Fabrica de chibrituri
nr. 9—11, București
Republica Socialistă România